# H8S/2345 Series

H8S/2345, H8S/2344, H8S/2343,
H8S/2341, H8S/2340

# H8S/2345 F-ZTAT™

Hardware Manual

HITACHI

ADE-602-129A
Rev. 2.0
1/12/98
Hitachi, Ltd.

Hitachi semiconductor

# Cautions

1. Hitachi neither warrants nor grants licenses of any rights of Hitachi's or any third party's patent, copyright, trademark, or other intellectual property rights for information contained in this document. Hitachi bears no responsibility for problems that may arise with third party's rights, including intellectual property rights, in connection with use of the information contained in this document.
2. Products and product specifications may be subject to change without notice. Confirm that you have received the latest product standards or specifications before final design, purchase or use.
3. Hitachi makes every attempt to ensure that its products are of high quality and reliability. However, contact Hitachi's sales office before using the product in an application that demands especially high quality and reliability or where its failure or malfunction may directly threaten human life or cause risk of bodily injury, such as aerospace, aeronautics, nuclear power, combustion control, transportation, traffic, safety equipment or medical equipment for life support.
4. Design your application so that the product is used within the ranges guaranteed by Hitachi particularly for maximum rating, operating supply voltage range, heat radiation characteristics, installation conditions and other characteristics. Hitachi bears no responsibility for failure or damage when used beyond the guaranteed ranges. Even within the guaranteed ranges, consider normally foreseeable failure rates or failure modes in semiconductor devices and employ systemic measures such as fail-safes, so that the equipment incorporating Hitachi product does not cause bodily injury, fire or other consequential damage due to operation of the Hitachi product.
5. This product is not designed to be radiation resistant.
6. No one is permitted to reproduce or duplicate, in any form, the whole or part of this document without written approval from Hitachi.
7. Contact Hitachi's sales office for any questions regarding this document or Hitachi semiconductor products.

# Main Amendments and Additions in this Edition

| Page | Item | Revision |
|---|---|---|
| Throughout | • H8S/2344, H8S/2341, and H8S/2340 added; F-ZTAT version of current H8S/2345 added. Generic name adopted: H8S/2345 Series, H8S/2345 F-ZTAT Hardware Manual.<br>• Notes added where necessary indicating that the H8S/2340 is a ROMless version, and only supports MCU operating modes 1, 4, and 5.<br>• Notes added where necessary indicating that the H8S/2345 F-ZTAT version only supports MCU operating modes 4 to 7, 10, 11, 14, and 15 (and that modes 1 to 3 (normal modes) cannot be used).<br>• Notes added where necessary indicating that the FWE pin applies only to the F-ZTAT version, and that this pin is $\overline{\text{WDTOVF}}$ in the ZTAT, mask ROM, and ROMless versions.<br>• Notes added where necessary indicating that the TFP-100G package is under development. | |
| 1 to 5 | 1.1 Overview | Amended (Information on newly added products) |
| 9 to 13 | Table 1.2 Pin Functions in Each Operating Mode | Amended<br>• PROM mode pin names partially changed<br>• Flash memory mode pin names added |
| 14 to 20 | Table 1.3 Pin Functions | Amended<br>• Addition of F-ZTAT version operating mode settings by pins MD2-MD0<br>• FWE pin description added |
| 69 to 72 | 3.1 Overview | Amended (Description of F-ZTAT and ROMless versions added) |
| 74 | System Control Register 2 (SYSCR2) (F-ZTAT Version Only) | New |
| 76 | 3.3.7 Mode 7 | Note 2 amended |
| 76, 77 | 3.3.8 Mode 8 to 3.3.13 Mode 15 | New |
| 78 | Table 3.3 Pin Functions in Each Mode | Amended (Mode 10, 11, 14, and 15 pin descriptions added) |
| 79 to 90 | 3.5 Memory Map in Each Operating Mode | Amended (Information on newly added products) |
| 107 | Table 5.3 Correspondence between Interrupt Sources and IPR Settings | Note amended |
| 141 | 6.2.5 Bus Control Register L (BCRL) | Description of bit 5 amended |

| Page | Item | Revision |
|---|---|---|
| 160 | Figure 6.14 Example of Wait Insertion Timing | Amended |
| 273 | 8.12.2 Register Configuration, Port G Data Direction Register (PGDDR) | Description amended |
| 294 to 309 | 9.2.3 Timer I/O Control Register (TIOR) | Amended (Register name added to tables) |
| 420 | 12.2.5 Serial Mode Register (SMR) | Description of bit 3 amended |
| 429 to 431 | Table 12.3 BRR Settings for Various Bit Rates (Asynchronous Mode) | Amendments to some Error column entries (values not entered for error of 3% or above) |
| 441 | Figure 12.2 Data Format in Asynchronous Communication (Example with 8-Bit Data, Parity, Two Stop Bits) | Amended |
| 461 | Figure 12.15 Sample SCI Initialization Flowchart | Note added |
| 467 | Figure 12.20 Sample Flowchart of Simultaneous Serial Transmit and Receive Operations | Note amended |
| 478, 479 | 13.2.2 Serial Status Register (SSR) | Description of bits 4 and 2 amended |
| 481 | 13.2.4 Serial Control Register (SCR) | Description of bits 1 and 0 amended |
| 483 | Figure 13.2 Schematic Diagram of Smart Card Interface Pin Connections | Amended |
| 484 | Figure 13.3 Smart Card Interface Data Format | Amended |
| 488, 489 | Table 13.5 Examples of Bit Rate B (bit/s) for Various BRR Settings (When n = 0)<br>Table 13.6 Examples of BRR Settings for Bit Rate B (bit/s) (When n = 0) | Amended (ø = 20.00 MHz column added) |
| 491 to 493 | 13.3.6 Data Transfer Operations, Serial Data Transmission | Amended |
| 497, 498 | 13.3.7 Operation in GSM Mode | Amended (Old section 13.3.7, Example of Use in Software Standby Mode, replaced with new section) |
| 510 | 14.2.3 A/D Control Register (ADCR) | Description of bits 7 and 6 amended |
| 519 to 524 | 14.6 Usage Notes | (1) Amendment of setting range for analog power supply pins etc.<br>(2) Deletion of module stop mode interrupts |
| 529 | 15.2.2 D/A Control Register (DACR) | Bit 5 description amended |
| 532 | 15.4 Usage Notes | New |

| Page | Item | Revision |
|---|---|---|
| 533 | 16.1 Overview | Description amended (Information on newly added products) |
| 534 | Figure 16.1 Block Diagram of RAM (H8S/2345, Advanced Mode) | Title of figure amended |
| 535 | 16.3 Operation | Description amended (Information on newly added products) |
| Whole of section 17 | Section 17 ROM<br>New flash memory description added, complete revision of section contents and layout | |
| Whole of section 20 | Section 20 Electrical Characteristics<br>Previous text used as electrical characteristics for ZTAT, mask ROM, and ROMless versions; new F-ZTAT version electrical characteristics added.<br>"Preliminary" notation deleted and "TBD" replaced with values for ZTAT, mask ROM, and ROMless versions. | |
| 666 | Figure 20.9 Reset Input Timing | Amended |
| 669 | Figure 20.12 Basic Bus Timing (Three-State Access) | Amended ($t_{WDS}$ specification) |
| 675 | Figure 20.24 SCK Clock Input Timing | Amended ($t_{SCKW}$ specification) |
| 677 to 752 | Appendix A Instruction Set | Amended (Replaced with latest version) |
| 753 to 759 | B.1 Addresses | Amended (Addition of registers used by F-ZTAT version) |
| 760 to 858 | B.2 Functions | Amended<br>• Addition of registers used by F-ZTAT version<br>• Amendment of note on interrupt priority registers A-K |
| 893 | Table F.1 H8S/2345 Series Product Code Lineup | Amended (Information on newly added products) |

# Preface

The H8S/2345 Series is a series of high-performance microcontrollers with a 32-bit H8S/2000 CPU core, and a set of on-chip supporting functions required for system configuration.

The H8S/2000 CPU can execute basic instructions in one state, and is provided with sixteen 16-bit general registers with a 32-bit internal configuration, and a concise and optimized instruction set. The CPU can handle a 16 Mbyte linear address space (architecturally 4 Gbytes). Programs based on the high-level language C can also be run efficiently.

The address space is divided into eight areas. The data bus width and access states can be selected for each of these areas, and various kinds of memory can be connected fast and easily.

On-chip memory consists of large-capacity ROM and RAM. With regard to on-chip ROM*[1], single power supply flash memory (F-ZTAT™*[2]), PROM (ZTAT™*[2]), and mask ROM versions are available, providing a quick and flexible response to conditions from ramp-up through full-scale volume production, even for applications with frequently changing specifications.

On-chip supporting functions include a 16-bit timer pulse unit (TPU), 8-bit timers, watchdog timer (WDT), serial communication interface (SCI), A/D converter, D/A converter, and I/O ports.

An on-chip data transfer controller (DTC) is also provided, enabling high-speed data transfer without CPU intervention.

Use of the H8S/2345 Series enables compact, high-performance systems to be implemented easily.

This manual describes the hardware of the H8S/2345 Series. Refer to the H8S/2600 Series and H8S/2000 Series Programming Manual for a detailed description of the instruction set.

Notes: 1. The H8S/2345, H8S/2344, H8S/2343, and H8S/2341 have on-chip ROM.
The H8S/2340 does not have on-chip ROM.

2. F-ZTAT (Flexible-ZTAT) is a trademark of Hitachi, Ltd.
ZTAT is a trademark of Hitachi, Ltd.

# Contents


Section 1 Overview ..... 1
1.1 Overview ..... 1
1.2 Block Diagram ..... 6
1.3 Pin Description ..... 7
1.3.1 Pin Arrangement ..... 7
1.3.2 Pin Functions in Each Operating Mode ..... 9
1.3.3 Pin Functions ..... 14

Section 2 CPU ..... 21
2.1 Overview ..... 21
2.1.1 Features ..... 21
2.1.2 Differences between H8S/2600 CPU and H8S/2000 CPU ..... 22
2.1.3 Differences from H8/300 CPU ..... 23
2.1.4 Differences from H8/300H CPU ..... 23
2.2 CPU Operating Modes ..... 24
2.3 Address Space ..... 29
2.4 Register Configuration ..... 30
2.4.1 Overview ..... 30
2.4.2 General Registers ..... 31
2.4.3 Control Registers ..... 32
2.4.4 Initial Register Values ..... 34
2.5 Data Formats ..... 35
2.5.1 General Register Data Formats ..... 35
2.5.2 Memory Data Formats ..... 37
2.6 Instruction Set ..... 38
2.6.1 Overview ..... 38
2.6.2 Instructions and Addressing Modes ..... 39
2.6.3 Table of Instructions Classified by Function ..... 41
2.6.4 Basic Instruction Formats ..... 51
2.7 Addressing Modes and Effective Address Calculation ..... 52
2.7.1 Addressing Mode ..... 52
2.7.2 Effective Address Calculation ..... 55
2.8 Processing States ..... 59
2.8.1 Overview ..... 59
2.8.2 Reset State ..... 60
2.8.3 Exception-Handling State ..... 61
2.8.4 Program Execution State ..... 64
2.8.5 Bus-Released State ..... 64
2.8.6 Power-Down State ..... 64

2.9 Basic Timing ..... 65
2.9.1 Overview ..... 65
2.9.2 On-Chip Memory (ROM, RAM) ..... 65
2.9.3 On-Chip Supporting Module Access Timing ..... 67
2.9.4 External Address Space Access Timing ..... 68

Section 3 MCU Operating Modes ..... 69
3.1 Overview ..... 69
3.1.1 Operating Mode Selection (F-ZTAT™ Version) ..... 69
3.1.2 Operating Mode Selection (ZTAT, Mask ROM, and No On-Chip ROM Versions) ..... 70
3.1.3 Register Configuration ..... 72
3.2 Register Descriptions ..... 72
3.2.1 Mode Control Register (MDCR) ..... 72
3.2.2 System Control Register (SYSCR) ..... 73
3.2.3 System Control Register 2 (SYSCR2) (F-ZTAT Version Only) ..... 74
3.3 Operating Mode Descriptions ..... 75
3.3.1 Mode 1 (ZTAT, Mask ROM, and No On-Chip ROM Versions Only) ..... 75
3.3.2 Mode 2*1 (ZTAT and Mask ROM Versions Only) ..... 75
3.3.3 Mode 3*1 (ZTAT and Mask ROM Versions Only) ..... 75
3.3.4 Mode 4*2 ..... 75
3.3.5 Mode 5*2 ..... 76
3.3.6 Mode 6*1 ..... 76
3.3.7 Mode 7*1 ..... 76
3.3.8 Modes 8 and 9 (F-ZTAT Version Only) ..... 76
3.3.9 Mode 10 (F-ZTAT Version Only) ..... 77
3.3.10 Mode 11 (F-ZTAT Version Only) ..... 77
3.3.11 Modes 12 and 13 (F-ZTAT Version Only) ..... 77
3.3.12 Mode 14 (F-ZTAT Version Only) ..... 77
3.3.13 Mode 15 (F-ZTAT Version Only) ..... 77
3.4 Pin Functions in Each Operating Mode ..... 78
3.5 Memory Map in Each Operating Mode ..... 79

Section 4 Exception Handling ..... 91
4.1 Overview ..... 91
4.1.1 Exception Handling Types and Priority ..... 91
4.1.2 Exception Handling Operation ..... 92
4.1.3 Exception Vector Table ..... 92
4.2 Reset ..... 94
4.2.1 Overview ..... 94
4.2.2 Reset Types ..... 94
4.2.3 Reset Sequence ..... 95
4.2.4 Interrupts after Reset ..... 96

4.2.5 State of On-Chip Supporting Modules after Reset Release ..... 96
4.3 Traces ..... 97
4.4 Interrupts ..... 98
4.5 Trap Instruction ..... 99
4.6 Stack Status after Exception Handling ..... 100
4.7 Notes on Use of the Stack ..... 101

Section 5 Interrupt Controller ..... 103
5.1 Overview ..... 103
5.1.1 Features ..... 103
5.1.2 Block Diagram ..... 104
5.1.3 Pin Configuration ..... 105
5.1.4 Register Configuration ..... 105
5.2 Register Descriptions ..... 106
5.2.1 System Control Register (SYSCR) ..... 106
5.2.2 Interrupt Priority Registers A to K (IPRA to IPRK) ..... 107
5.2.3 IRQ Enable Register (IER) ..... 108
5.2.4 IRQ Sense Control Registers H and L (ISCRH, ISCRL) ..... 109
5.2.5 IRQ Status Register (ISR) ..... 110
5.3 Interrupt Sources ..... 111
5.3.1 External Interrupts ..... 111
5.3.2 Internal Interrupts ..... 112
5.3.3 Interrupt Exception Handling Vector Table ..... 112
5.4 Interrupt Operation ..... 116
5.4.1 Interrupt Control Modes and Interrupt Operation ..... 116
5.4.2 Interrupt Control Mode 0 ..... 119
5.4.3 Interrupt Control Mode 2 ..... 121
5.4.4 Interrupt Exception Handling Sequence ..... 123
5.4.5 Interrupt Response Times ..... 125
5.5 Usage Notes ..... 126
5.5.1 Contention between Interrupt Generation and Disabling ..... 126
5.5.2 Instructions that Disable Interrupts ..... 127
5.5.3 Times when Interrupts are Disabled ..... 127
5.5.4 Interrupts during Execution of EEPMOV Instruction ..... 127
5.6 DTC Activation by Interrupt ..... 128
5.6.1 Overview ..... 128
5.6.2 Block Diagram ..... 128
5.6.3 Operation ..... 129

Section 6 Bus Controller ..... 131
6.1 Overview ..... 131
6.1.1 Features ..... 131
6.1.2 Block Diagram ..... 132

6.1.3 Pin Configuration ..... 133
6.1.4 Register Configuration ..... 133
6.2 Register Descriptions ..... 134
6.2.1 Bus Width Control Register (ABWCR) ..... 134
6.2.2 Access State Control Register (ASTCR) ..... 135
6.2.3 Wait Control Registers H and L (WCRH, WCRL) ..... 136
6.2.4 Bus Control Register H (BCRH) ..... 139
6.2.5 Bus Control Register L (BCRL) ..... 141
6.3 Overview of Bus Control ..... 142
6.3.1 Area Partitioning ..... 142
6.3.2 Bus Specifications ..... 144
6.3.3 Memory Interfaces ..... 145
6.3.4 Advanced Mode ..... 145
6.3.5 Areas in Normal Mode (ZTAT, Mask ROM, and No On-Chip ROM Versions Only) ..... 146
6.3.6 Chip Select Signals ..... 147
6.4 Basic Bus Interface ..... 148
6.4.1 Overview ..... 148
6.4.2 Data Size and Data Alignment ..... 148
6.4.3 Valid Strobes ..... 150
6.4.4 Basic Timing ..... 151
6.4.5 Wait Control ..... 159
6.5 Burst ROM Interface ..... 161
6.5.1 Overview ..... 161
6.5.2 Basic Timing ..... 161
6.5.3 Wait Control ..... 163
6.6 Idle Cycle ..... 164
6.6.1 Operation ..... 164
6.6.2 Pin States in Idle Cycle ..... 167
6.7 Bus Release ..... 167
6.7.1 Overview ..... 167
6.7.2 Operation ..... 167
6.7.3 Pin States in External Bus Released State ..... 168
6.7.4 Transition Timing ..... 169
6.7.5 Usage Note ..... 170
6.8 Bus Arbitration ..... 170
6.8.1 Overview ..... 170
6.8.2 Operation ..... 170
6.8.3 Bus Transfer Timing ..... 171
6.8.4 External Bus Release Usage Note ..... 171
6.9 Resets and the Bus Controller ..... 171

Section 7 Data Transfer Controller ..... 173
7.1 Overview ..... 173
7.1.1 Features ..... 173
7.1.2 Block Diagram ..... 174
7.1.3 Register Configuration ..... 175
7.2 Register Descriptions ..... 176
7.2.1 DTC Mode Register A (MRA) ..... 176
7.2.2 DTC Mode Register B (MRB) ..... 178
7.2.3 DTC Source Address Register (SAR) ..... 179
7.2.4 DTC Destination Address Register (DAR) ..... 179
7.2.5 DTC Transfer Count Register A (CRA) ..... 179
7.2.6 DTC Transfer Count Register B (CRB) ..... 180
7.2.7 DTC Enable Registers (DTCER) ..... 180
7.2.8 DTC Vector Register (DTVECR) ..... 181
7.2.9 Module Stop Control Register (MSTPCR) ..... 182
7.3 Operation ..... 183
7.3.1 Overview ..... 183
7.3.2 Activation Sources ..... 185
7.3.3 DTC Vector Table ..... 186
7.3.4 Location of Register Information in Address Space ..... 189
7.3.5 Normal Mode ..... 190
7.3.6 Repeat Mode ..... 191
7.3.7 Block Transfer Mode ..... 192
7.3.8 Chain Transfer ..... 194
7.3.9 Operation Timing ..... 195
7.3.10 Number of DTC Execution States ..... 196
7.3.11 Procedures for Using DTC ..... 198
7.3.12 Examples of Use of the DTC ..... 199
7.4 Interrupts ..... 201
7.5 Usage Notes ..... 201

Section 8 I/O Ports ..... 203
8.1 Overview ..... 203
8.2 Port 1 ..... 208
8.2.1 Overview ..... 208
8.2.2 Register Configuration ..... 209
8.2.3 Pin Functions ..... 210
8.3 Port 2 ..... 219
8.3.1 Overview ..... 219
8.3.2 Register Configuration ..... 219
8.3.3 Pin Functions ..... 221
8.4 Port 3 ..... 230
8.4.1 Overview ..... 230

8.4.2 Register Configuration ..... 230
8.4.3 Pin Functions ..... 233
8.5 Port 4 ..... 235
8.5.1 Overview ..... 235
8.5.2 Register Configuration ..... 236
8.5.3 Pin Functions ..... 236
8.6 Port A ..... 237
8.6.1 Overview ..... 237
8.6.2 Register Configuration ..... 238
8.6.3 Pin Functions ..... 241
8.6.4 MOS Input Pull-Up Function ..... 242
8.7 Port B ..... 243
8.7.1 Overview ..... 243
8.7.2 Register Configuration ..... 244
8.7.3 Pin Functions ..... 246
8.7.4 MOS Input Pull-Up Function ..... 248
8.8 Port C ..... 249
8.8.1 Overview ..... 249
8.8.2 Register Configuration ..... 250
8.8.3 Pin Functions ..... 252
8.8.4 MOS Input Pull-Up Function ..... 254
8.9 Port D ..... 255
8.9.1 Overview ..... 255
8.9.2 Register Configuration ..... 256
8.9.3 Pin Functions ..... 258
8.9.4 MOS Input Pull-Up Function ..... 259
8.10 Port E ..... 260
8.10.1 Overview ..... 260
8.10.2 Register Configuration ..... 261
8.10.3 Pin Functions ..... 263
8.10.4 MOS Input Pull-Up Function ..... 264
8.11 Port F ..... 265
8.11.1 Overview ..... 265
8.11.2 Register Configuration ..... 266
8.11.3 Pin Functions ..... 269
8.12 Port G ..... 271
8.12.1 Overview ..... 271
8.12.2 Register Configuration ..... 272
8.12.3 Pin Functions ..... 275

Section 9 16-Bit Timer Pulse Unit (TPU) ..... 277
9.1 Overview ..... 277
9.1.1 Features ..... 277

9.1.2 Block Diagram ... 281
9.1.3 Pin Configuration ... 282
9.1.4 Register Configuration ... 284
9.2 Register Descriptions ... 286
9.2.1 Timer Control Register (TCR) ... 286
9.2.2 Timer Mode Register (TMDR) ... 291
9.2.3 Timer I/O Control Register (TIOR) ... 293
9.2.4 Timer Interrupt Enable Register (TIER) ... 310
9.2.5 Timer Status Register (TSR) ... 313
9.2.6 Timer Counter (TCNT) ... 316
9.2.7 Timer General Register (TGR) ... 317
9.2.8 Timer Start Register (TSTR) ... 318
9.2.9 Timer Synchro Register (TSYR) ... 319
9.2.10 Module Stop Control Register (MSTPCR) ... 320
9.3 Interface to Bus Master ... 321
9.3.1 16-Bit Registers ... 321
9.3.2 8-Bit Registers ... 321
9.4 Operation ... 323
9.4.1 Overview ... 323
9.4.2 Basic Functions ... 324
9.4.3 Synchronous Operation ... 330
9.4.4 Buffer Operation ... 332
9.4.5 Cascaded Operation ... 336
9.4.6 PWM Modes ... 338
9.4.7 Phase Counting Mode ... 343
9.5 Interrupts ... 349
9.5.1 Interrupt Sources and Priorities ... 349
9.5.2 DTC Activation ... 351
9.5.3 A/D Converter Activation ... 351
9.6 Operation Timing ... 352
9.6.1 Input/Output Timing ... 352
9.6.2 Interrupt Signal Timing ... 356
9.7 Usage Notes ... 360

Section 10 8-Bit Timers ... 371
10.1 Overview ... 371
10.1.1 Features ... 371
10.1.2 Block Diagram ... 372
10.1.3 Pin Configuration ... 373
10.1.4 Register Configuration ... 373
10.2 Register Descriptions ... 374
10.2.1 Timer Counters 0 and 1 (TCNT0, TCNT1) ... 374
10.2.2 Time Constant Registers A0 and A1 (TCORA0, TCORA1) ... 374

10.2.3 Time Constant Registers B0 and B1 (TCORB0, TCORB1) ..... 375
10.2.4 Time Control Registers 0 and 1 (TCR0, TCR1) ..... 375
10.2.5 Timer Control/Status Registers 0 and 1 (TCSR0, TCSR1) ..... 377
10.2.6 Module Stop Control Register (MSTPCR) ..... 380
10.3 Operation ..... 381
10.3.1 TCNT Incrementation Timing ..... 381
10.3.2 Compare Match Timing ..... 382
10.3.3 Timing of External RESET on TCNT ..... 384
10.3.4 Timing of Overflow Flag (OVF) Setting ..... 384
10.3.5 Operation with Cascaded Connection ..... 385
10.4 Interrupts ..... 386
10.4.1 Interrupt Sources and DTC Activation ..... 386
10.4.2 A/D Converter Activation ..... 386
10.5 Sample Application ..... 387
10.6 Usage Notes ..... 388
10.6.1 Contention between TCNT Write and Clear ..... 388
10.6.2 Contention between TCNT Write and Increment ..... 389
10.6.3 Contention between TCOR Write and Compare Match ..... 390
10.6.4 Contention between Compare Matches A and B ..... 391
10.6.5 Switching of Internal Clocks and TCNT Operation ..... 391
10.6.6 Usage Note ..... 393

Section 11 Watchdog Timer ..... 395
11.1 Overview ..... 395
11.1.1 Features ..... 395
11.1.2 Block Diagram ..... 396
11.1.3 Pin Configuration ..... 397
11.1.4 Register Configuration ..... 397
11.2 Register Descriptions ..... 398
11.2.1 Timer Counter (TCNT) ..... 398
11.2.2 Timer Control/Status Register (TCSR) ..... 398
11.2.3 Reset Control/Status Register (RSTCSR) ..... 400
11.2.4 Notes on Register Access ..... 402
11.3 Operation ..... 404
11.3.1 Watchdog Timer Operation ..... 404
11.3.2 Interval Timer Operation ..... 406
11.3.3 Timing of Setting Overflow Flag (OVF) ..... 406
11.3.4 Timing of Setting of Watchdog Timer Overflow Flag (WOVF) ..... 407
11.4 Interrupts ..... 408
11.5 Usage Notes ..... 408
11.5.1 Contention between Timer Counter (TCNT) Write and Increment ..... 408
11.5.2 Changing Value of CKS2 to CKS0 ..... 408
11.5.3 Switching between Watchdog Timer Mode and Interval Timer Mode ..... 409

11.5.4 System Reset by $\overline{\text{WDTOVF}}$ Signal ..... 409
11.5.5 Internal Reset in Watchdog Timer Mode ..... 409

Section 12 Serial Communication Interface (SCI) ..... 411
12.1 Overview ..... 411
12.1.1 Features ..... 411
12.1.2 Block Diagram ..... 413
12.1.3 Pin Configuration ..... 414
12.1.4 Register Configuration ..... 415
12.2 Register Descriptions ..... 416
12.2.1 Receive Shift Register (RSR) ..... 416
12.2.2 Receive Data Register (RDR) ..... 416
12.2.3 Transmit Shift Register (TSR) ..... 417
12.2.4 Transmit Data Register (TDR) ..... 417
12.2.5 Serial Mode Register (SMR) ..... 418
12.2.6 Serial Control Register (SCR) ..... 421
12.2.7 Serial Status Register (SSR) ..... 425
12.2.8 Bit Rate Register (BRR) ..... 428
12.2.9 Smart Card Mode Register (SCMR) ..... 437
12.2.10 Module Stop Control Register (MSTPCR) ..... 438
12.3 Operation ..... 439
12.3.1 Overview ..... 439
12.3.2 Operation in Asynchronous Mode ..... 441
12.3.3 Multiprocessor Communication Function ..... 452
12.3.4 Operation in Clocked Synchronous Mode ..... 460
12.4 SCI Interrupts ..... 468
12.5 Usage Notes ..... 469

Section 13 Smart Card Interface ..... 473
13.1 Overview ..... 473
13.1.1 Features ..... 473
13.1.2 Block Diagram ..... 474
13.1.3 Pin Configuration ..... 475
13.1.4 Register Configuration ..... 476
13.2 Register Descriptions ..... 477
13.2.1 Smart Card Mode Register (SCMR) ..... 477
13.2.2 Serial Status Register (SSR) ..... 478
13.2.3 Serial Mode Register (SMR) ..... 480
13.2.4 Serial Control Register (SCR) ..... 481
13.3 Operation ..... 482
13.3.1 Overview ..... 482
13.3.2 Pin Connections ..... 482
13.3.3 Data Format ..... 484

13.3.4 Register Settings ..... 486
13.3.5 Clock ..... 488
13.3.6 Data Transfer Operations ..... 490
13.3.7 Operation in GSM Mode ..... 497
13.4 Usage Note ..... 498

Section 14 A/D Converter ..... 503
14.1 Overview ..... 503
14.1.1 Features ..... 503
14.1.2 Block Diagram ..... 504
14.1.3 Pin Configuration ..... 505
14.1.4 Register Configuration ..... 506
14.2 Register Descriptions ..... 507
14.2.1 A/D Data Registers A to D (ADDRA to ADDRD) ..... 507
14.2.2 A/D Control/Status Register (ADCSR) ..... 508
14.2.3 A/D Control Register (ADCR) ..... 510
14.2.4 Module Stop Control Register (MSTPCR) ..... 511
14.3 Interface to Bus Master ..... 512
14.4 Operation ..... 513
14.4.1 Single Mode (SCAN = 0) ..... 513
14.4.2 Scan Mode (SCAN = 1) ..... 515
14.4.3 Input Sampling and A/D Conversion Time ..... 517
14.4.4 External Trigger Input Timing ..... 518
14.5 Interrupts ..... 519
14.6 Usage Notes ..... 519

Section 15 D/A Converter ..... 525
15.1 Overview ..... 525
15.1.1 Features ..... 525
15.1.2 Block Diagram ..... 526
15.1.3 Pin Configuration ..... 527
15.1.4 Register Configuration ..... 527
15.2 Register Descriptions ..... 528
15.2.1 D/A Data Registers 0 and 1 (DADR0, DADR1) ..... 528
15.2.2 D/A Control Register (DACR) ..... 528
15.2.3 Module Stop Control Register (MSTPCR) ..... 530
15.3 Operation ..... 531
15.4 Usage Notes ..... 532

Section 16 RAM ..... 533
16.1 Overview ..... 533
16.1.1 Block Diagram ..... 534
16.1.2 Register Configuration ..... 534

16.2 Register Descriptions ..... 535
16.2.1 System Control Register (SYSCR) ..... 535
16.3 Operation ..... 535
16.4 Usage Note ..... 535

Section 17 ROM ..... 537
17.1 Overview ..... 537
17.1.1 Block Diagram ..... 537
17.1.2 Register Configuration ..... 538
17.2 Register Descriptions ..... 538
17.2.1 Mode Control Register (MDCR) ..... 538
17.2.2 Bus Control Register L (BCRL) ..... 539
17.3 Operation ..... 539
17.4 PROM Mode ..... 542
17.4.1 PROM Mode Setting ..... 542
17.4.2 Socket Adapter and Memory Map ..... 542
17.5 Programming ..... 545
17.5.1 Overview ..... 545
17.5.2 Programming and Verification ..... 545
17.5.3 Programming Precautions ..... 549
17.5.4 Reliability of Programmed Data ..... 550
17.6 Overview of Flash Memory ..... 551
17.6.1 Features ..... 551
17.6.2 Block Diagram ..... 552
17.6.3 Flash Memory Operating Modes ..... 553
17.6.4 Pin Configuration ..... 558
17.6.5 Register Configuration ..... 559
17.7 Register Descriptions ..... 560
17.7.1 Flash Memory Control Register 1 (FLMCR1) ..... 560
17.7.2 Flash Memory Control Register 2 (FLMCR2) ..... 563
17.7.3 Erase Block Registers 1 and 2 (EBR1, EBR2) ..... 564
17.7.4 System Control Register 2 (SYSCR2) ..... 565
17.7.5 RAM Emulation Register (RAMER) ..... 566
17.8 On-Board Programming Modes ..... 568
17.8.1 Boot Mode ..... 569
17.8.2 User Program Mode ..... 573
17.9 Programming/Erasing Flash Memory ..... 575
17.9.1 Program Mode — Preliminary — ..... 576
17.9.2 Program-Verify Mode — Preliminary — ..... 577
17.9.3 Erase Mode — Preliminary — ..... 579
17.9.4 Erase-Verify Mode — Preliminary — ..... 579
17.10 Flash Memory Protection ..... 581
17.10.1 Hardware Protection ..... 581

17.10.2 Software Protection ..... 581
17.10.3 Error Protection ..... 582
17.11 Flash Memory Emulation in RAM ..... 585
17.11.1 Emulation in RAM ..... 585
17.11.2 RAM Overlap ..... 586
17.12 Interrupt Handling when Programming/Erasing Flash Memory ..... 587
17.13 Flash Memory Writer Mode ..... 588
17.13.1 Writer Mode Setting ..... 588
17.13.2 Socket Adapters and Memory Map ..... 589
17.13.3 Writer Mode Operation ..... 590
17.13.4 Memory Read Mode ..... 592
17.13.5 Auto-Program Mode ..... 596
17.13.6 Auto-Erase Mode ..... 598
17.13.7 Status Read Mode ..... 599
17.13.8 Status Polling ..... 601
17.13.9 Writer Mode Transition Time ..... 602
17.13.10 Notes On Memory Programming ..... 602
17.14 Flash Memory Programming and Erasing Precautions ..... 603

Section 18 Clock Pulse Generator ..... 609
18.1 Overview ..... 609
18.1.1 Block Diagram ..... 609
18.1.2 Register Configuration ..... 609
18.2 Register Descriptions ..... 610
18.2.1 System Clock Control Register (SCKCR) ..... 610
18.3 Oscillator ..... 611
18.3.1 Connecting a Crystal Resonator ..... 611
18.3.2 External Clock Input ..... 613
18.4 Duty Adjustment Circuit ..... 615
18.5 Medium-Speed Clock Divider ..... 615
18.6 Bus Master Clock Selection Circuit ..... 615

Section 19 Power-Down Modes ..... 617
19.1 Overview ..... 617
19.1.1 Register Configuration ..... 618
19.2 Register Descriptions ..... 619
19.2.1 Standby Control Register (SBYCR) ..... 619
19.2.2 System Clock Control Register (SCKCR) ..... 620
19.2.3 Module Stop Control Register (MSTPCR) ..... 621
19.3 Medium-Speed Mode ..... 622
19.4 Sleep Mode ..... 623
19.5 Module Stop Mode ..... 623
19.5.1 Module Stop Mode ..... 623

19.5.2 Usage Notes ..... 624
19.6 Software Standby Mode ..... 625
19.6.1 Software Standby Mode ..... 625
19.6.2 Clearing Software Standby Mode ..... 625
19.6.3 Setting Oscillation Stabilization Time after Clearing Software Standby Mode ..... 626
19.6.4 Software Standby Mode Application Example ..... 626
19.6.5 Usage Notes ..... 627
19.7 Hardware Standby Mode ..... 628
19.7.1 Hardware Standby Mode ..... 628
19.7.2 Hardware Standby Mode Timing ..... 628
19.8 ø Clock Output Disabling Function ..... 629

Section 20 Electrical Characteristics ..... 631
20.1 Electrical Characteristics of F-ZTAT Version ..... 631
20.1.1 Absolute Maximum Ratings ..... 631
20.1.2 DC Characteristics ..... 632
20.1.3 AC Characteristics ..... 639
20.1.4 A/D Conversion Characteristics ..... 646
20.1.5 D/A Conversion Characteristics ..... 647
20.1.6 Flash Memory Characteristics ..... 648
20.2 Electrical Characteristics of ZTAT, Mask ROM, and No On-chip ROM Versions ..... 650
20.2.1 Absolute Maximum Ratings ..... 650
20.2.2 DC Characteristics ..... 651
20.2.3 AC Characteristics ..... 656
20.2.4 A/D Conversion Characteristics ..... 663
20.2.5 D/A Conversion Characteristics ..... 664
20.3 Operation Timing ..... 665
20.3.1 Clock Timing ..... 665
20.3.2 Control Signal Timing ..... 666
20.3.3 Bus Timing ..... 667
20.3.4 Timing for On-Chip Supporting Modules ..... 673
20.4 Usage Note ..... 676

Appendix A Instruction Set ..... 677
A.1 Instruction List ..... 677
A.2 Instruction Codes ..... 701
A.3 Operation Code Map ..... 715
A.4 Number of States Required for Instruction Execution ..... 719
A.5 Bus States During Instruction Execution ..... 733
A.6 Condition Code Modification ..... 747

Appendix B Internal I/O Register ..... 753
B.1 Addresses ..... 753
B.2 Functions ..... 760

Appendix C I/O Port Block Diagrams ..... 859
C.1 Port 1 Block Diagram ..... 859
C.2 Port 2 Block Diagram ..... 863
C.3 Port 3 Block Diagram ..... 867
C.4 Port 4 Block Diagram ..... 870
C.5 Port A Block Diagram ..... 871
C.6 Port B Block Diagram ..... 872
C.7 Port C Block Diagram ..... 873
C.8 Port D Block Diagram ..... 874
C.9 Port E Block Diagram ..... 875
C.10 Port F Block Diagram ..... 876
C.11 Port G Block Diagram ..... 884

Appendix D Pin States ..... 888
D.1 Port States in Each Mode ..... 888

Appendix E Timing of Transition to and Recovery from Hardware Standby Mode ..... 892

Appendix F Product Code Lineup ..... 893

Appendix G Package Dimensions ..... 894

# Section 1 Overview

## 1.1 Overview

The H8S/2345 Series is a series of microcomputers (MCUs: microcomputer units), built around the H8S/2000 CPU, employing Hitachi's proprietary architecture, and equipped with peripheral functions on-chip.

The H8S/2000 CPU has an internal 32-bit architecture, is provided with sixteen 16-bit general registers and a concise, optimized instruction set designed for high-speed operation, and can address a 16-Mbyte linear address space. The instruction set is upward-compatible with H8/300 and H8/300H CPU instructions at the object-code level, facilitating migration from the H8/300, H8/300L, or H8/300H Series.

On-chip peripheral functions required for system configuration include data transfer controller (DTC) bus masters, ROM and RAM memory, a16-bit timer-pulse unit (TPU), 8-bit timer, watchdog timer (WDT), serial communication interface (SCI), A/D converter, D/A converter, and I/O ports.

The on-chip ROM*[1] is either single power supply flash memory (F-ZTAT™*[2]), PROM (ZTAT™*[2]), or mask ROM, with a capacity of 128, 96, 64, or 32 kbytes. ROM is connected to the CPU via a 16-bit data bus, enabling both byte and word data to be accessed in one state. Instruction fetching has been speeded up, and processing speed increased.

Seven operating modes, modes 1 to 7, are provided, and there is a choice of address space and single-chip mode or external expansion mode.

The features of the H8S/2345 Series are shown in Table 1.1.

Notes: 1. The H8S/2345, H8S/2344, H8S/2343, and H8S/2341 have on-chip ROM. The H8S/2340 does not have on-chip ROM.

2. F-ZTAT™ is a trademark of Hitachi, Ltd.
   ZTAT is a trademark of Hitachi, Ltd.

**Table 1.1 Overview**

| Item | Specification |
|---|---|
| CPU | • General-register machine<br>— Sixteen 16-bit general registers (also usable as sixteen 8-bit registers or eight 32-bit registers)<br>• High-speed operation suitable for realtime control<br>— Maximum clock rate: 20 MHz<br>— High-speed arithmetic operations<br>8/16/32-bit register-register add/subtract: 50 ns<br>16 × 16-bit register-register multiply : 1000 ns<br>32 ÷ 16-bit register-register divide : 1000 ns<br>• Instruction set suitable for high-speed operation<br>— Sixty-five basic instructions<br>— 8/16/32-bit move/arithmetic and logic instructions<br>— Unsigned/signed multiply and divide instructions<br>— Powerful bit-manipulation instructions<br>• Two CPU operating modes<br>— Normal mode: 64-kbyte address space (ZTAT, mask ROM, and ROMless versions only)<br>— Advanced mode: 16-Mbyte address space |
| Bus controller | • Address space divided into 8 areas, with bus specifications settable independently for each area<br>• Chip select output possible for areas 0 to 3<br>• Choice of 8-bit or 16-bit access space for each area<br>• 2-state or 3-state access space can be designated for each area<br>• Number of program wait states can be set for each area<br>• Burst ROM directly connectable<br>• External bus release function |
| Data transfer controller (DTC) | • Can be activated by internal interrupt or software<br>• Multiple transfers or multiple types of transfer possible for one activation source<br>• Transfer is possible in repeat mode, block transfer mode, etc.<br>• Request can be sent to CPU for interrupt that activated DTC |
| 16-bit timer-pulse unit (TPU) | • 6-channel 16-bit timer on-chip<br>• Pulse I/O processing capability for up to 16 pins'<br>• Automatic 2-phase encoder count capability |

**Table 1.1 Overview (cont)**

| Item | Specification |
|---|---|
| 8-bit timer<br>2 channels | • 8-bit up-counter (external event count capability)<br>• Two time constant registers<br>• Two-channel connection possible |
| Watchdog timer | • Watchdog timer or interval timer selectable |
| Serial communication interface (SCI)<br>2 channels | • Asynchronous mode or synchronous mode selectable<br>• Multiprocessor communication function<br>• Smart card interface function |
| A/D converter | • Resolution: 10 bits<br>• Input: 8 channels<br>• High-speed conversion: 6.7 μs minimum conversion time (at 20 MHz operation)<br>• Single or scan mode selectable<br>• Sample and hold circuit<br>• A/D conversion can be activated by external trigger or timer trigger |
| D/A converter | • Resolution: 8 bits<br>• Output: 2 channels |
| I/O ports | • 71 I/O pins, 8 input-only pins |
| Memory | • Flash memory, PROM, or mask ROM<br>• High-speed static RAM |

| Product Name | ROM | RAM |
|---|---|---|
| H8S/2345 | 128 kbytes | 4 kbytes |
| H8S/2344 | 96 kbytes | 4 kbytes |
| H8S/2343 | 64 kbytes | 2 kbytes |
| H8S/2341 | 32 kbytes | 2 kbytes |
| H8S/2340 | — | 2 kbytes |

| | |
|---|---|
| Interrupt controller | • Nine external interrupt pins (NMI, $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ7}}$)<br>• 43 internal interrupt sources<br>• Eight priority levels settable |

**Table 1.1 Overview (cont)**

| Item | Specification |
|---|---|
| Power-down state | • Medium-speed mode<br>• Sleep mode<br>• Module stop mode<br>• Software standby mode<br>• Hardware standby mode |
| Operating modes | • Eight MCU operating modes (F-ZTAT version) |

| Mode | CPU Operating Mode | Description | On-Chip ROM | External Data Bus: Initial Value | External Data Bus: Maximum Value |
|---|---|---|---|---|---|
| 0 | — | — | — | — | — |
| 1 | | | | | |
| 2 | | | | | |
| 3 | | | | | |
| 4 | Advanced | On-chip ROM disabled expansion mode | Disabled | 16 bits | 16 bits |
| 5 | | | | 8 bits | 16 bits |
| 6 | | On-chip ROM enabled expansion mode | Enabled | 8 bits | 16 bits |
| 7 | | Single-chip mode | | — | — |
| 8 | — | — | — | — | — |
| 9 | | | | | |
| 10 | Advanced | Boot mode | Enabled | 8 bits | 16 bits |
| 11 | | | | — | — |
| 12 | — | — | — | — | — |
| 13 | | | | | |
| 14 | Advanced | User-programmable mode | Enabled | 8 bits | 16 bits |
| 15 | | | | — | — |

**Table 1.1 Overview (cont)**

| Item | Specification |
|---|---|

**Operating modes**

- Seven MCU operating modes (ZTAT, mask ROM, and ROMless versions)

| Mode | CPU Operating Mode | Description | On-Chip ROM | External Data Bus: Initial Value | External Data Bus: Maximum Value |
|---|---|---|---|---|---|
| 1 | Normal | On-chip ROM disabled expansion mode | Disabled | 8 bits | 16 bits |
| 2* | | On-chip ROM enabled expansion mode | Enabled | 8 bits | 16 bits |
| 3* | | Single-chip mode | Enabled | — | |
| 4 | Advanced | On-chip ROM disabled expansion mode | Disabled | 16 bits | 16 bits |
| 5 | | On-chip ROM disabled expansion mode | Disabled | 8 bits | 16 bits |
| 6* | | On-chip ROM enabled expansion mode | Enabled | 8 bits | 16 bits |
| 7* | | Single-chip mode | Enabled | — | |

Note: * Not used on ROMless versions.

**Clock pulse generator**

- Built-in duty correction circuit

**Packages**

- 100-pin plastic TQFP (TFP-100B, TFP-100G*)
- 100-pin plastic QFP (FP-100A, FP-100B)

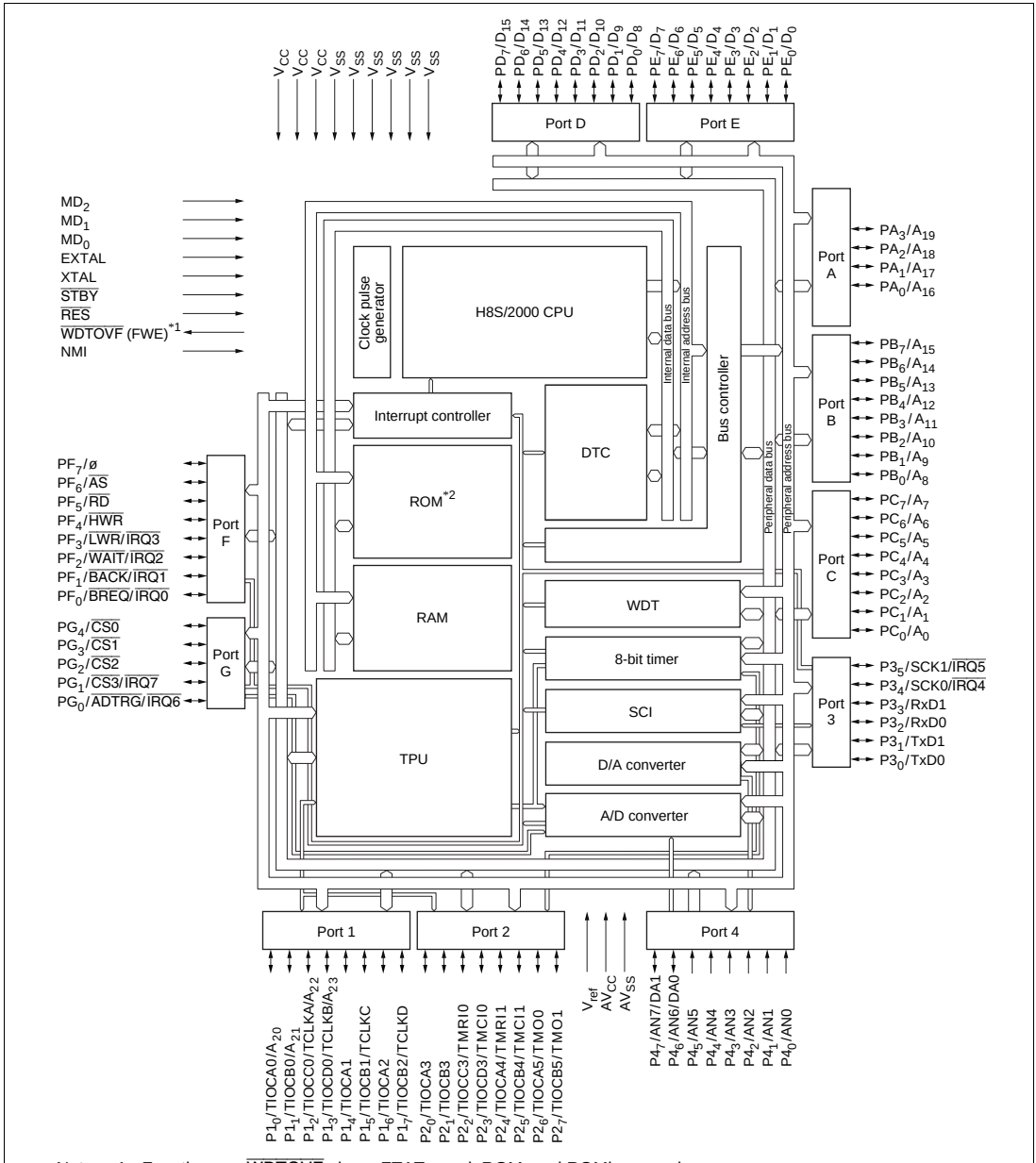
**Product lineup**

| Model Name: Mask ROM Version | F-ZTAT™ | ZTAT™ | ROM/RAM (Bytes) | Packages |
|---|---|---|---|---|
| HD6432345 | HD64F2345 | HD6472345 | 128 k/4 k | TFP-100B |
| HD6432344 | — | — | 96 k/4 k | TFP-100G* |
| HD6432343 | — | — | 64 k/2 k | FP-100A |
| HD6432341 | — | — | 32 k/2 k | FP-100B |
| HD6412340 (ROMless versions) | — | — | —/2 k | |

Note: * TFP-100G is under development.

## 1.2 Block Diagram

Figure 1.1 shows an internal block diagram of the H8S/2345 Series.



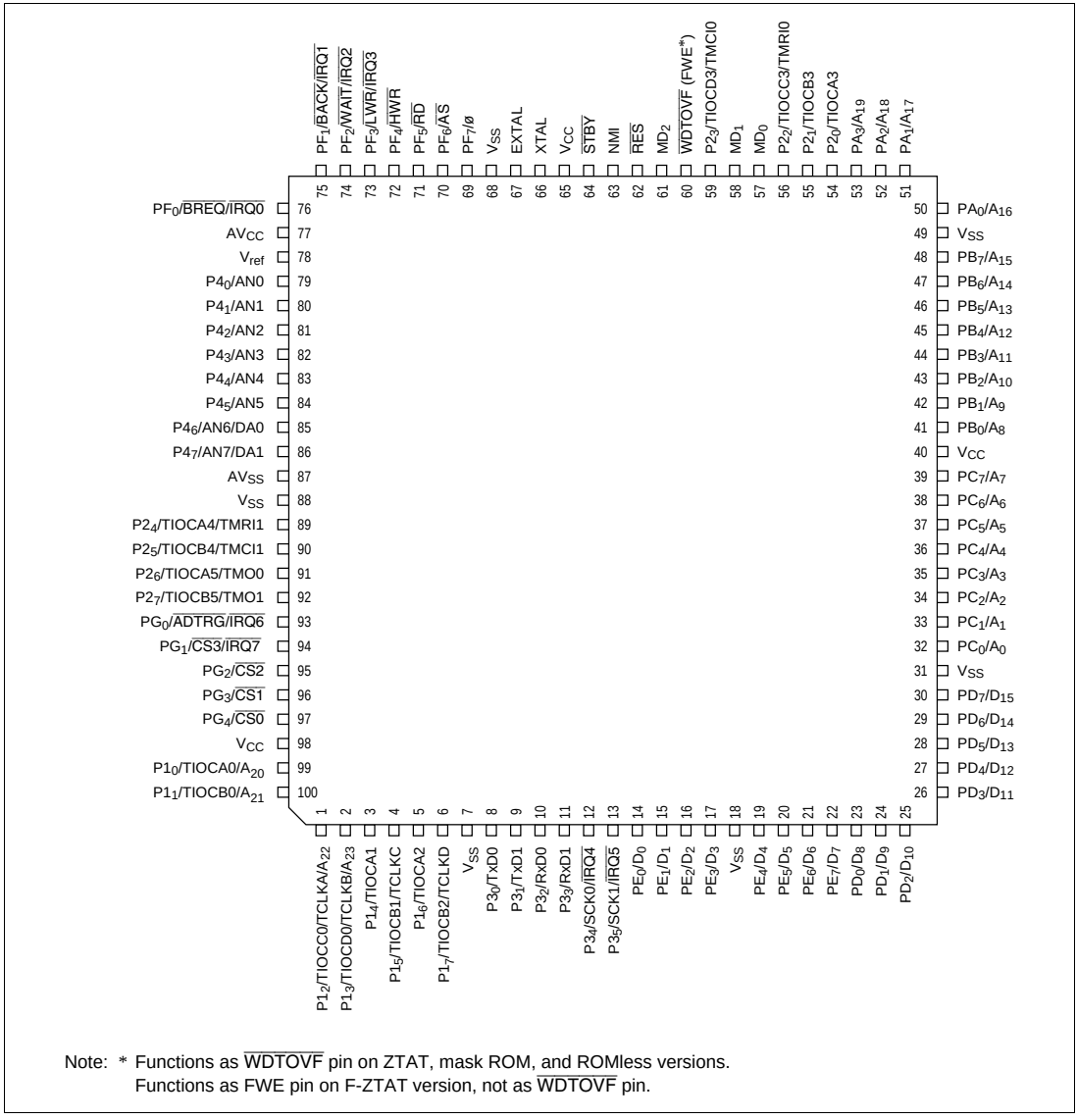

Notes: 1. Functions as $\overline{\text{WDTOVF}}$ pin on ZTAT, mask ROM, and ROMless versions.
Functions as FWE pin on F-ZTAT version, not as $\overline{\text{WDTOVF}}$ pin.
2. Not present on ROMless version.

**Figure 1.1 Block Diagram**

## 1.3 Pin Description

### 1.3.1 Pin Arrangement

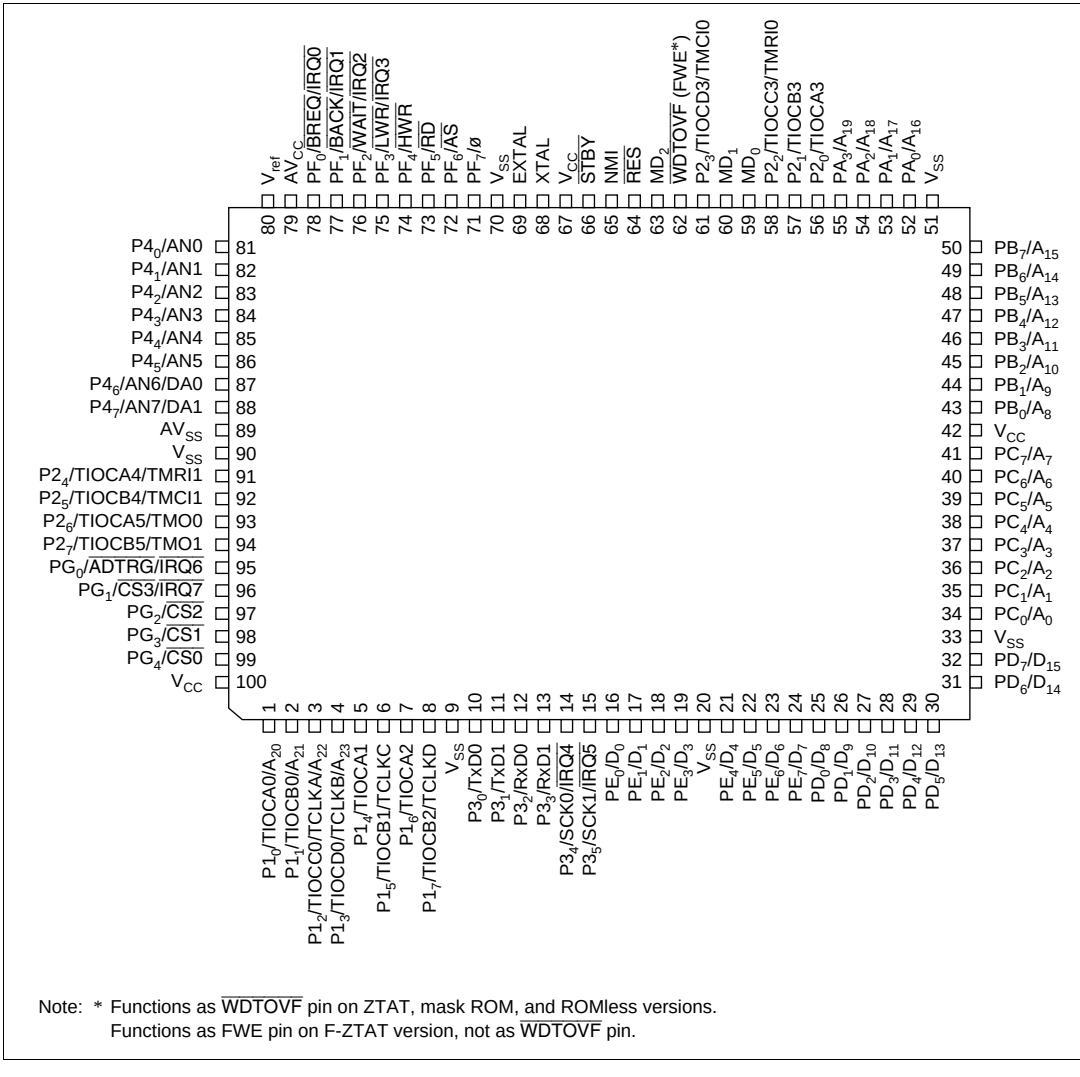
Figures 1.2 and 1.3 show the pin arrangement of the H8S/2345 Series.




Note: * Functions as WDTOVF pin on ZTAT, mask ROM, and ROMless versions.
Functions as FWE pin on F-ZTAT version, not as WDTOVF pin.

**Figure 1.2 Pin Arrangement (FP-100B, TFP-100B, TFP-100G*: Top View)**

Note: TFP-100G is under development.




Note: * Functions as WDTOVF pin on ZTAT, mask ROM, and ROMless versions.
Functions as FWE pin on F-ZTAT version, not as WDTOVF pin.

**Figure 1.3 Pin Arrangement (FP-100A: Top View)**

### 1.3.2 Pin Functions in Each Operating Mode

Table 1.2 shows the pin functions of the H8S/2345 Series in each of the operating modes.

**Table 1.2 Pin Functions in Each Operating Mode**

| Pin No. | | Pin Name | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| FP-100B, TFP-100B, TFP-100G | FP-100A | Mode 1*1 | Mode 2*1, *2 | Mode 3*1, *2 | Mode 4 | Mode 5 | Mode 6*2 | Mode 7*2 | PROM Mode*3 | Flash Memory Writer Mode*4 |
| 1 | 3 | $P1_2$/ TIOCC0/ TCLKA | $P1_2$/ TIOCC0/ TCLKA | $P1_2$/ TIOCC0/ TCLKA | $P1_2$/ TIOCC0/ TCLKA/ $A_{22}$ | $P1_2$/ TIOCC0/ TCLKA/ $A_{22}$ | $P1_2$/ TIOCC0/ TCLKA/ $A_{22}$ | $P1_2$/ TIOCC0/ TCLKA | NC | NC |
| 2 | 4 | $P1_3$/ TIOCD0/ TCLKB | $P1_3$/ TIOCD0/ TCLKB | $P1_3$/ TIOCD0/ TCLKB | $P1_3$/ TIOCD0/ TCLKB/ $A_{23}$ | $P1_3$/ TIOCD0/ TCLKB/ $A_{23}$ | $P1_3$/ TIOCD0/ TCLKB/ $A_{23}$ | $P1_3$/ TIOCD0/ TCLKB | NC | NC |
| 3 | 5 | $P1_4$/ TIOCA1 | $P1_4$/ TIOCA1 | $P1_4$/ TIOCA1 | $P1_4$/ TIOCA1 | $P1_4$/ TIOCA1 | $P1_4$/ TIOCA1 | $P1_4$/ TIOCA1 | NC | NC |
| 4 | 6 | $P1_5$/ TIOCB1/ TCLKC | $P1_5$/ TIOCB1/ TCLKC | $P1_5$/ TIOCB1/ TCLKC | $P1_5$/ TIOCB1/ TCLKC | $P1_5$/ TIOCB1/ TCLKC | $P1_5$/ TIOCB1/ TCLKC | $P1_5$/ TIOCB1/ TCLKC | NC | NC |
| 5 | 7 | $P1_6$/ TIOCA2 | $P1_6$/ TIOCA2 | $P1_6$/ TIOCA2 | $P1_6$/ TIOCA2 | $P1_6$/ TIOCA2 | $P1_6$/ TIOCA2 | $P1_6$/ TIOCA2 | NC | NC |
| 6 | 8 | $P1_7$/ TIOCB2/ TCLKD | $P1_7$/ TIOCB2/ TCLKD | $P1_7$/ TIOCB2/ TCLKD | $P1_7$/ TIOCB2/ TCLKD | $P1_7$/ TIOCB2/ TCLKD | $P1_7$/ TIOCB2/ TCLKD | $P1_7$/ TIOCB2/ TCLKD | NC | NC |
| 7 | 9 | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ |
| 8 | 10 | $P3_0$/TxD0 | $P3_0$/TxD0 | $P3_0$/TxD0 | $P3_0$/TxD0 | $P3_0$/TxD0 | $P3_0$/TxD0 | $P3_0$/TxD0 | NC | NC |
| 9 | 11 | $P3_1$/TxD1 | $P3_1$/TxD1 | $P3_1$/TxD1 | $P3_1$/TxD1 | $P3_1$/TxD1 | $P3_1$/TxD1 | $P3_1$/TxD1 | NC | NC |
| 10 | 12 | $P3_2$/RxD0 | $P3_2$/RxD0 | $P3_2$/RxD0 | $P3_2$/RxD0 | $P3_2$/RxD0 | $P3_2$/RxD0 | $P3_2$/RxD0 | NC | NC |
| 11 | 13 | $P3_3$/RxD1 | $P3_3$/RxD1 | $P3_3$/RxD1 | $P3_3$/RxD1 | $P3_3$/RxD1 | $P3_3$/RxD1 | $P3_3$/RxD1 | NC | NC |
| 12 | 14 | $P3_4$/ SCK0/ $\overline{IRQ4}$ | $P3_4$/ SCK0/ $\overline{IRQ4}$ | $P3_4$/ SCK0/ $\overline{IRQ4}$ | $P3_4$/ SCK0/ $\overline{IRQ4}$ | $P3_4$/ SCK0/ $\overline{IRQ4}$ | $P3_4$/ SCK0/ $\overline{IRQ4}$ | $P3_4$/ SCK0/ $\overline{IRQ4}$ | NC | NC |
| 13 | 15 | $P3_5$/ SCK1/ $\overline{IRQ5}$ | $P3_5$/ SCK1/ $\overline{IRQ5}$ | $P3_5$/ SCK1/ $\overline{IRQ5}$ | $P3_5$/ SCK1/ $\overline{IRQ5}$ | $P3_5$/ SCK1/ $\overline{IRQ5}$ | $P3_5$/ SCK1/ $\overline{IRQ5}$ | $P3_5$/ SCK1/ $\overline{IRQ5}$ | NC | NC |
| 14 | 16 | $PE_0$/$D_0$ | $PE_0$/$D_0$ | $PE_0$ | $PE_0$/$D_0$ | $PE_0$/$D_0$ | $PE_0$/$D_0$ | $PE_0$ | NC | NC |
| 15 | 17 | $PE_1$/$D_1$ | $PE_1$/$D_1$ | $PE_1$ | $PE_1$/$D_1$ | $PE_1$/$D_1$ | $PE_1$/$D_1$ | $PE_1$ | NC | NC |
| 16 | 18 | $PE_2$/$D_2$ | $PE_2$/$D_2$ | $PE_2$ | $PE_2$/$D_2$ | $PE_2$/$D_2$ | $PE_2$/$D_2$ | $PE_2$ | NC | NC |
| 17 | 19 | $PE_3$/$D_3$ | $PE_3$/$D_3$ | $PE_3$ | $PE_3$/$D_3$ | $PE_3$/$D_3$ | $PE_3$/$D_3$ | $PE_3$ | NC | NC |
| 18 | 20 | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ |
| 19 | 21 | $PE_4$/$D_4$ | $PE_4$/$D_4$ | $PE_4$ | $PE_4$/$D_4$ | $PE_4$/$D_4$ | $PE_4$/$D_4$ | $PE_4$ | NC | NC |

## Table 1.2 Pin Functions in Each Operating Mode (cont)

| Pin No. | | Pin Name | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| FP-100B, TFP-100B, TFP-100G | FP-100A | Mode 1*1 | Mode 2*1, *2 | Mode 3*1, *2 | Mode 4 | Mode 5 | Mode 6*2 | Mode 7*2 | PROM Mode*3 | Flash Memory Writer Mode*4 |
| 20 | 22 | $PE_5/D_5$ | $PE_5/D_5$ | $PE_5$ | $PE_5/D_5$ | $PE_5/D_5$ | $PE_5/D_5$ | $PE_5$ | NC | NC |
| 21 | 23 | $PE_6/D_6$ | $PE_6/D_6$ | $PE_6$ | $PE_6/D_6$ | $PE_6/D_6$ | $PE_6/D_6$ | $PE_6$ | NC | NC |
| 22 | 24 | $PE_7/D_7$ | $PE_7/D_7$ | $PE_7$ | $PE_7/D_7$ | $PE_7/D_7$ | $PE_7/D_7$ | $PE_7$ | NC | NC |
| 23 | 25 | $D_8$ | $D_8$ | $PD_0$ | $D_8$ | $D_8$ | $D_8$ | $PD_0$ | $EO_0$ | $FO_0$ |
| 24 | 26 | $D_9$ | $D_9$ | $PD_1$ | $D_9$ | $D_9$ | $D_9$ | $PD_1$ | $EO_1$ | $FO_1$ |
| 25 | 27 | $D_{10}$ | $D_{10}$ | $PD_2$ | $D_{10}$ | $D_{10}$ | $D_{10}$ | $PD_2$ | $EO_2$ | $FO_2$ |
| 26 | 28 | $D_{11}$ | $D_{11}$ | $PD_3$ | $D_{11}$ | $D_{11}$ | $D_{11}$ | $PD_3$ | $EO_3$ | $FO_3$ |
| 27 | 29 | $D_{12}$ | $D_{12}$ | $PD_4$ | $D_{12}$ | $D_{12}$ | $D_{12}$ | $PD_4$ | $EO_4$ | $FO_4$ |
| 28 | 30 | $D_{13}$ | $D_{13}$ | $PD_5$ | $D_{13}$ | $D_{13}$ | $D_{13}$ | $PD_5$ | $EO_5$ | $FO_5$ |
| 29 | 31 | $D_{14}$ | $D_{14}$ | $PD_6$ | $D_{14}$ | $D_{14}$ | $D_{14}$ | $PD_6$ | $EO_6$ | $FO_6$ |
| 30 | 32 | $D_{15}$ | $D_{15}$ | $PD_7$ | $D_{15}$ | $D_{15}$ | $D_{15}$ | $PD_7$ | $EO_7$ | $FO_7$ |
| 31 | 33 | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ |
| 32 | 34 | $A_0$ | $PC_0/A_0$ | $PC_0$ | $A_0$ | $A_0$ | $PC_0/A_0$ | $PC_0$ | $EA_0$ | $FA_0$ |
| 33 | 35 | $A_1$ | $PC_1/A_1$ | $PC_1$ | $A_1$ | $A_1$ | $PC_1/A_1$ | $PC_1$ | $EA_1$ | $FA_1$ |
| 34 | 36 | $A_2$ | $PC_2/A_2$ | $PC_2$ | $A_2$ | $A_2$ | $PC_2/A_2$ | $PC_2$ | $EA_2$ | $FA_2$ |
| 35 | 37 | $A_3$ | $PC_3/A_3$ | $PC_3$ | $A_3$ | $A_3$ | $PC_3/A_3$ | $PC_3$ | $EA_3$ | $FA_3$ |
| 36 | 38 | $A_4$ | $PC_4/A_4$ | $PC_4$ | $A_4$ | $A_4$ | $PC_4/A_4$ | $PC_4$ | $EA_4$ | $FA_4$ |
| 37 | 39 | $A_5$ | $PC_5/A_5$ | $PC_5$ | $A_5$ | $A_5$ | $PC_5/A_5$ | $PC_5$ | $EA_5$ | $FA_5$ |
| 38 | 40 | $A_6$ | $PC_6/A_6$ | $PC_6$ | $A_6$ | $A_6$ | $PC_6/A_6$ | $PC_6$ | $EA_6$ | $FA_6$ |
| 39 | 41 | $A_7$ | $PC_7/A_7$ | $PC_7$ | $A_7$ | $A_7$ | $PC_7/A_7$ | $PC_7$ | $EA_7$ | $FA_7$ |
| 40 | 42 | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ |
| 41 | 43 | $A_8$ | $PB_0/A_8$ | $PB_0$ | $A_8$ | $A_8$ | $PB_0/A_8$ | $PB_0$ | $EA_8$ | $FA_8$ |
| 42 | 44 | $A_9$ | $PB_1/A_9$ | $PB_1$ | $A_9$ | $A_9$ | $PB_1/A_9$ | $PB_1$ | $\overline{OE}$ | $FA_9$ |
| 43 | 45 | $A_{10}$ | $PB_2/A_{10}$ | $PB_2$ | $A_{10}$ | $A_{10}$ | $PB_2/A_{10}$ | $PB_2$ | $EA_{10}$ | $FA_{10}$ |
| 44 | 46 | $A_{11}$ | $PB_3/A_{11}$ | $PB_3$ | $A_{11}$ | $A_{11}$ | $PB_3/A_{11}$ | $PB_3$ | $EA_{11}$ | $FA_{11}$ |
| 45 | 47 | $A_{12}$ | $PB_4/A_{12}$ | $PB_4$ | $A_{12}$ | $A_{12}$ | $PB_4/A_{12}$ | $PB_4$ | $EA_{12}$ | $FA_{12}$ |
| 46 | 48 | $A_{13}$ | $PB_5/A_{13}$ | $PB_5$ | $A_{13}$ | $A_{13}$ | $PB_5/A_{13}$ | $PB_5$ | $EA_{13}$ | $FA_{13}$ |
| 47 | 49 | $A_{14}$ | $PB_6/A_{14}$ | $PB_6$ | $A_{14}$ | $A_{14}$ | $PB_6/A_{14}$ | $PB_6$ | $EA_{14}$ | $FA_{14}$ |
| 48 | 50 | $A_{15}$ | $PB_7/A_{15}$ | $PB_7$ | $A_{15}$ | $A_{15}$ | $PB_7/A_{15}$ | $PB_7$ | $EA_{15}$ | $FA_{15}$ |
| 49 | 51 | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ |
| 50 | 52 | $PA_0$ | $PA_0$ | $PA_0$ | $A_{16}$ | $A_{16}$ | $PA_0/A_{16}$ | $PA_0$ | $EA_{16}$ | $FA_{16}$ |
| 51 | 53 | $PA_1$ | $PA_1$ | $PA_1$ | $A_{17}$ | $A_{17}$ | $PA_1/A_{17}$ | $PA_1$ | $V_{CC}$ | NC |

## Table 1.2 Pin Functions in Each Operating Mode (cont)

| Pin No. | | Pin Name | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **FP-100B, TFP-100B, TFP-100G** | **FP-100A** | **Mode 1***1 | **Mode 2***1, *2 | **Mode 3***1, *2 | **Mode 4** | **Mode 5** | **Mode 6***2 | **Mode 7***2 | **PROM Mode***3 | **Flash Memory Writer Mode***4 |
| 52 | 54 | $PA_2$ | $PA_2$ | $PA_2$ | $A_{18}$ | $A_{18}$ | $PA_2/A_{18}$ | $PA_2$ | $V_{CC}$ | NC |
| 53 | 55 | $PA_3$ | $PA_3$ | $PA_3$ | $A_{19}$ | $A_{19}$ | $PA_3/A_{19}$ | $PA_3$ | NC | NC |
| 54 | 56 | $P2_0$/ TIOCA3 | $P2_0$/ TIOCA3 | $P2_0$/ TIOCA3 | $P2_0$/ TIOCA3 | $P2_0$/ TIOCA3 | $P2_0$/ TIOCA3 | $P2_0$/ TIOCA3 | NC | $\overline{OE}$ |
| 55 | 57 | $P2_1$/ TIOCB3 | $P2_1$/ TIOCB3 | $P2_1$/ TIOCB3 | $P2_1$/ TIOCB3 | $P2_1$/ TIOCB3 | $P2_1$/ TIOCB3 | $P2_1$/ TIOCB3 | NC | $\overline{CE}$ |
| 56 | 58 | $P2_2$/ TIOCC3/ TMRI0 | $P2_2$/ TIOCC3/ TMRI0 | $P2_2$/ TIOCC3/ TMRI0 | $P2_2$/ TIOCC3/ TMRI0 | $P2_2$/ TIOCC3/ TMRI0 | $P2_2$/ TIOCC3/ TMRI0 | $P2_2$/ TIOCC3/ TMRI0 | NC | $\overline{WE}$ |
| 57 | 59 | $MD_0$ | $MD_0$ | $MD_0$ | $MD_0$ | $MD_0$ | $MD_0$ | $MD_0$ | $V_{SS}$ | $V_{SS}$ |
| 58 | 60 | $MD_1$ | $MD_1$ | $MD_1$ | $MD_1$ | $MD_1$ | $MD_1$ | $MD_1$ | $V_{SS}$ | $V_{SS}$ |
| 59 | 61 | $P2_3$/ TIOCD3/ TMCI0 | $P2_3$/ TIOCD3/ TMCI0 | $P2_3$/ TIOCD3/ TMCI0 | $P2_3$/ TIOCD3/ TMCI0 | $P2_3$/ TIOCD3/ TMCI0 | $P2_3$/ TIOCD3/ TMCI0 | $P2_3$/ TIOCD3/ TMCI0 | NC | $V_{CC}$ |
| 60 | 62 | $\overline{WDTOVF}$ | $\overline{WDTOVF}$ | $\overline{WDTOVF}$ | $\overline{WDTOVF}$ ($FWE$*5) | $\overline{WDTOVF}$ ($FWE$*5) | $\overline{WDTOVF}$ ($FWE$*5) | $\overline{WDTOVF}$ ($FWE$*5) | NC | FWE |
| 61 | 63 | $MD_2$ | $MD_2$ | $MD_2$ | $MD_2$ | $MD_2$ | $MD_2$ | $MD_2$ | $V_{SS}$ | $V_{SS}$ |
| 62 | 64 | $\overline{RES}$ | $\overline{RES}$ | $\overline{RES}$ | $\overline{RES}$ | $\overline{RES}$ | $\overline{RES}$ | $\overline{RES}$ | $V_{PP}$ | $\overline{RES}$ |
| 63 | 65 | NMI | NMI | NMI | NMI | NMI | NMI | NMI | $EA_9$ | $V_{CC}$ |
| 64 | 66 | $\overline{STBY}$ | $\overline{STBY}$ | $\overline{STBY}$ | $\overline{STBY}$ | $\overline{STBY}$ | $\overline{STBY}$ | $\overline{STBY}$ | $V_{SS}$ | $V_{CC}$ |
| 65 | 67 | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ |
| 66 | 68 | XTAL | XTAL | XTAL | XTAL | XTAL | XTAL | XTAL | NC | XTAL |
| 67 | 69 | EXTAL | EXTAL | EXTAL | EXTAL | EXTAL | EXTAL | EXTAL | NC | EXTAL |
| 68 | 70 | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ |
| 69 | 71 | $PF_7$/ø | $PF_7$/ø | $PF_7$/ø | $PF_7$/ø | $PF_7$/ø | $PF_7$/ø | $PF_7$/ø | NC | NC |
| 70 | 72 | $\overline{AS}$ | $\overline{AS}$ | $PF_6$ | $\overline{AS}$ | $\overline{AS}$ | $\overline{AS}$ | $PF_6$ | NC | NC |
| 71 | 73 | $\overline{RD}$ | $\overline{RD}$ | $PF_5$ | $\overline{RD}$ | $\overline{RD}$ | $\overline{RD}$ | $PF_5$ | NC | NC |
| 72 | 74 | $\overline{HWR}$ | $\overline{HWR}$ | $PF_4$ | $\overline{HWR}$ | $\overline{HWR}$ | $\overline{HWR}$ | $PF_4$ | NC | NC |
| 73 | 75 | $\overline{LWR}$ | $\overline{LWR}$ | $PF_3/\overline{IRQ3}$ | $\overline{LWR}$ | $\overline{LWR}$ | $\overline{LWR}$ | $PF_3/\overline{IRQ3}$ | NC | NC |
| 74 | 76 | $PF_2$/ $\overline{WAIT}$/ $\overline{IRQ2}$ | $PF_2$/ $\overline{WAIT}$/ $\overline{IRQ2}$ | $PF_2/\overline{IRQ2}$ | $PF_2$/ $\overline{WAIT}$/ $\overline{IRQ2}$ | $PF_2$/ $\overline{WAIT}$/ $\overline{IRQ2}$ | $PF_2$/ $\overline{WAIT}$/ $\overline{IRQ2}$ | $PF_2/\overline{IRQ2}$ | $\overline{CE}$ | $V_{CC}$ |
| 75 | 77 | $PF_1$/ $\overline{BACK}$/ $\overline{IRQ1}$ | $PF_1$/ $\overline{BACK}$/ $\overline{IRQ1}$ | $PF_1/\overline{IRQ1}$ | $PF_1$/ $\overline{BACK}$/ $\overline{IRQ1}$ | $PF_1$/ $\overline{BACK}$/ $\overline{IRQ1}$ | $PF_1$/ $\overline{BACK}$/ $\overline{IRQ1}$ | $PF_1/\overline{IRQ1}$ | $\overline{PGM}$ | $V_{SS}$ |

## Table 1.2 Pin Functions in Each Operating Mode (cont)

| Pin No. | | Pin Name | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| FP-100B, TFP-100B, TFP-100G | FP-100A | Mode 1*1 | Mode 2*1, *2 | Mode 3*1, *2 | Mode 4 | Mode 5 | Mode 6*2 | Mode 7*2 | PROM Mode*3 | Flash Memory Writer Mode*4 |
| 76 | 78 | $PF_0$/$\overline{BREQ}$/$\overline{IRQ0}$ | $PF_0$/$\overline{BREQ}$/$\overline{IRQ0}$ | $PF_0$/$\overline{IRQ0}$ | $PF_0$/$\overline{BREQ}$/$\overline{IRQ0}$ | $PF_0$/$\overline{BREQ}$/$\overline{IRQ0}$ | $PF_0$/$\overline{BREQ}$/$\overline{IRQ0}$ | $PF_0$/$\overline{IRQ0}$ | NC | $V_{SS}$ |
| 77 | 79 | $AV_{CC}$ | $AV_{CC}$ | $AV_{CC}$ | $AV_{CC}$ | $AV_{CC}$ | $AV_{CC}$ | $AV_{CC}$ | $V_{CC}$ | $V_{CC}$ |
| 78 | 80 | $V_{ref}$ | $V_{ref}$ | $V_{ref}$ | $V_{ref}$ | $V_{ref}$ | $V_{ref}$ | $V_{ref}$ | $V_{CC}$ | $V_{CC}$ |
| 79 | 81 | $P4_0$/AN0 | $P4_0$/AN0 | $P4_0$/AN0 | $P4_0$/AN0 | $P4_0$/AN0 | $P4_0$/AN0 | $P4_0$/AN0 | NC | NC |
| 80 | 82 | $P4_1$/AN1 | $P4_1$/AN1 | $P4_1$/AN1 | $P4_1$/AN1 | $P4_1$/AN1 | $P4_1$/AN1 | $P4_1$/AN1 | NC | NC |
| 81 | 83 | $P4_2$/AN2 | $P4_2$/AN2 | $P4_2$/AN2 | $P4_2$/AN2 | $P4_2$/AN2 | $P4_2$/AN2 | $P4_2$/AN2 | NC | NC |
| 82 | 84 | $P4_3$/AN3 | $P4_3$/AN3 | $P4_3$/AN3 | $P4_3$/AN3 | $P4_3$/AN3 | $P4_3$/AN3 | $P4_3$/AN3 | NC | NC |
| 83 | 85 | $P4_4$/AN4 | $P4_4$/AN4 | $P4_4$/AN4 | $P4_4$/AN4 | $P4_4$/AN4 | $P4_4$/AN4 | $P4_4$/AN4 | NC | NC |
| 84 | 86 | $P4_5$/AN5 | $P4_5$/AN5 | $P4_5$/AN5 | $P4_5$/AN5 | $P4_5$/AN5 | $P4_5$/AN5 | $P4_5$/AN5 | NC | NC |
| 85 | 87 | $P4_6$/AN6/DA0 | $P4_6$/AN6/DA0 | $P4_6$/AN6/DA0 | $P4_6$/AN6/DA0 | $P4_6$/AN6/DA0 | $P4_6$/AN6/DA0 | $P4_6$/AN6/DA0 | NC | NC |
| 86 | 88 | $P4_7$/AN7/DA1 | $P4_7$/AN7/DA1 | $P4_7$/AN7/DA1 | $P4_7$/AN7/DA1 | $P4_7$/AN7/DA1 | $P4_7$/AN7/DA1 | $P4_7$/AN7/DA1 | NC | NC |
| 87 | 89 | $AV_{SS}$ | $AV_{SS}$ | $AV_{SS}$ | $AV_{SS}$ | $AV_{SS}$ | $AV_{SS}$ | $AV_{SS}$ | $V_{SS}$ | $V_{SS}$ |
| 88 | 90 | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ |
| 89 | 91 | $P2_4$/TIOCA4/TMRI1 | $P2_4$/TIOCA4/TMRI1 | $P2_4$/TIOCA4/TMRI1 | $P2_4$/TIOCA4/TMRI1 | $P2_4$/TIOCA4/TMRI1 | $P2_4$/TIOCA4/TMRI1 | $P2_4$/TIOCA4/TMRI1 | NC | NC |
| 90 | 92 | $P2_5$/TIOCB4/TMCI1 | $P2_5$/TIOCB4/TMCI1 | $P2_5$/TIOCB4/TMCI1 | $P2_5$/TIOCB4/TMCI1 | $P2_5$/TIOCB4/TMCI1 | $P2_5$/TIOCB4/TMCI1 | $P2_5$/TIOCB4/TMCI1 | NC | $V_{CC}$ |
| 91 | 93 | $P2_6$/TIOCA5/TMO0 | $P2_6$/TIOCA5/TMO0 | $P2_6$/TIOCA5/TMO0 | $P2_6$/TIOCA5/TMO0 | $P2_6$/TIOCA5/TMO0 | $P2_6$/TIOCA5/TMO0 | $P2_6$/TIOCA5/TMO0 | NC | NC |
| 92 | 94 | $P2_7$/TIOCB5/TMO1 | $P2_7$/TIOCB5/TMO1 | $P2_7$/TIOCB5/TMO1 | $P2_7$/TIOCB5/TMO1 | $P2_7$/TIOCB5/TMO1 | $P2_7$/TIOCB5/TMO1 | $P2_7$/TIOCB5/TMO1 | NC | NC |
| 93 | 95 | $PG_0$/$\overline{IRQ6}$/$\overline{ADTRG}$ | $PG_0$/$\overline{IRQ6}$/$\overline{ADTRG}$ | $PG_0$/$\overline{IRQ6}$/$\overline{ADTRG}$ | $PG_0$/$\overline{IRQ6}$/$\overline{ADTRG}$ | $PG_0$/$\overline{IRQ6}$/$\overline{ADTRG}$ | $PG_0$/$\overline{IRQ6}$/$\overline{ADTRG}$ | $PG_0$/$\overline{IRQ6}$/$\overline{ADTRG}$ | NC | NC |
| 94 | 96 | $PG_1$/$\overline{IRQ7}$ | $PG_1$/$\overline{IRQ7}$ | $PG_1$/$\overline{IRQ7}$ | $PG_1$/$\overline{CS3}$/$\overline{IRQ7}$ | $PG_1$/$\overline{CS3}$/$\overline{IRQ7}$ | $PG_1$/$\overline{CS3}$/$\overline{IRQ7}$ | $PG_1$/$\overline{IRQ7}$ | NC | NC |
| 95 | 97 | $PG_2$ | $PG_2$ | $PG_2$ | $PG_2$/$\overline{CS2}$ | $PG_2$/$\overline{CS2}$ | $PG_2$/$\overline{CS2}$ | $PG_2$ | NC | NC |
| 96 | 98 | $PG_3$ | $PG_3$ | $PG_3$ | $PG_3$/$\overline{CS1}$ | $PG_3$/$\overline{CS1}$ | $PG_3$/$\overline{CS1}$ | $PG_3$ | NC | NC |
| 97 | 99 | $PG_4$/$\overline{CS0}$ | $PG_4$/$\overline{CS0}$ | $PG_4$ | $PG_4$/$\overline{CS0}$ | $PG_4$/$\overline{CS0}$ | $PG_4$/$\overline{CS0}$ | $PG_4$ | NC | NC |

## Table 1.2 Pin Functions in Each Operating Mode (cont)

| Pin No. | | Pin Name | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **FP-100B, TFP-100B, TFP-100G** | **FP-100A** | **Mode 1**[*1] | **Mode 2**[*1, *2] | **Mode 3**[*1, *2] | **Mode 4** | **Mode 5** | **Mode 6**[*2] | **Mode 7**[*2] | **PROM Mode**[*3] | **Flash Memory Writer Mode**[*4] |
| 98 | 100 | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ | $V_{CC}$ |
| 99 | 1 | $P1_0$/ TIOCA0 | $P1_0$/ TIOCA0 | $P1_0$/ TIOCA0 | $P1_0$/ TIOCA0/ $A_{20}$ | $P1_0$/ TIOCA0/ $A_{20}$ | $P1_0$/ TIOCA0/ $A_{20}$ | $P1_0$/ TIOCA0 | NC | NC |
| 100 | 2 | $P1_1$/ TIOCB0 | $P1_1$/ TIOCB0 | $P1_1$/ TIOCB0 | $P1_1$/ TIOCB0/ $A_{21}$ | $P1_1$/ TIOCB0/ $A_{21}$ | $P1_1$/ TIOCB0/ $A_{21}$ | $P1_1$/ TIOCB0 | NC | NC |

Notes: 1. Modes 1 to 3 are not available on the F-ZTAT version.
2. Modes 2, 3, 6, and 7 are not available on the ROMless version.
3. ZTAT version only.
4. F-ZTAT version only.
5. The FWE pin is only used on the F-ZTAT version. It cannot be used as a $\overline{\text{WDTOVF}}$ pin on the F-ZTAT version.

### 1.3.3 Pin Functions

Table 1.3 outlines the pin functions of the H8S/2345 Series.

**Table 1.3 Pin Functions**

| | | Pin No. | | | |
|---|---|---|---|---|---|
| **Type** | **Symbol** | **FP-100B, TFP-100B, TFP-100G** | **FP-100A** | **I/O** | **Name and Function** |
| Power | $V_{CC}$ | 40, 65, 98 | 42, 67, 100 | Input | Power supply: For connection to the power supply. All $V_{CC}$ pins should be connected to the system power supply. |
| | $V_{SS}$ | 7, 18, 31, 49, 68, 88 | 9, 20, 33, 51, 70, 90 | Input | Ground: For connection to ground (0 V). All $V_{SS}$ pins should be connected to the system power supply (0 V). |
| Clock | XTAL | 66 | 68 | Input | Connects to a crystal oscillator. See section 18, Clock Pulse Generator, for typical connection diagrams for a crystal oscillator and external clock input. |
| | EXTAL | 67 | 69 | Input | Connects to a crystal oscillator. The EXTAL pin can also input an external clock. See section 18, Clock Pulse Generator, for typical connection diagrams for a crystal oscillator and external clock input. |
| | ø | 69 | 71 | Output | System clock: Supplies the system clock to an external device. |

**Table 1.3 Pin Functions (cont)**

| Type | Symbol | Pin No. FP-100B, TFP-100B, TFP-100G | Pin No. FP-100A | I/O | Name and Function |
|---|---|---|---|---|---|
| Operating mode control | $MD_2$ to $MD_0$ | 61, 58, 57 | 63, 60, 59 | Input | Mode pins: These pins set the operating mode.<br>The relation between the settings of pins $MD_2$ to $MD_0$ and the operating mode is shown below. These pins should not be changed while the H8S/2345 Series is operating.<br>• F-ZTAT Version |

| FWE | $MD_2$ | $MD_1$ | $MD_0$ | Operating Mode |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | — |
| | | | 1 | — |
| | | 1 | 0 | — |
| | | | 1 | — |
| | 1 | 0 | 0 | Mode 4 |
| | | | 1 | Mode 5 |
| | | 1 | 0 | Mode 6 |
| | | | 1 | Mode 7 |
| 1 | 0 | 0 | 0 | — |
| | | | 1 | — |
| | | 1 | 0 | Mode 10 |
| | | | 1 | Mode 11 |
| | 1 | 0 | 0 | — |
| | | | 1 | — |
| | | 1 | 0 | Mode 14 |
| | | | 1 | Mode 15 |

**Table 1.3 Pin Functions (cont)**

| Type | Symbol | Pin No. FP-100B, TFP-100B, TFP-100G | Pin No. FP-100A | I/O | Name and Function |
|---|---|---|---|---|---|
| Operating mode control | $MD_2$ to $MD_0$ | 61, 58, 57 | 63, 60, 59 | Input | • ZTAT, mask ROM, and ROMless versions<br>$MD_2$ $MD_1$ $MD_0$ Operating Mode<br>0 0 0 —<br>0 0 1 Mode 1<br>0 1 0 Mode 2*<br>0 1 1 Mode 3*<br>1 0 0 Mode 4<br>1 0 1 Mode 5<br>1 1 0 Mode 6*<br>1 1 1 Mode 7*<br>Note: * Not used on ROMless version. |
| System control | $\overline{RES}$ | 62 | 64 | Input | Reset input: When this pin is driven low, the chip is reset. The type of reset can be selected according to the NMI input level. At power-on, the NMI pin input level should be set high. |
| | $\overline{STBY}$ | 64 | 66 | Input | Standby: When this pin is driven low, a transition is made to hardware standby mode. |
| | $\overline{BREQ}$ | 76 | 78 | Input | Bus request: Used by an external bus master to issue a bus request to the H8S/2345 Series. |
| | $\overline{BACK}$ | 75 | 77 | Output | Bus request acknowledge: Indicates that the bus has been released to an external bus master. |
| | FWE*[1] | 60 | 62 | Input | Flash write enable: Enables or disables writing to flash memory. |

**Table 1.3 Pin Functions (cont)**

| Type | Symbol | Pin No. FP-100B, TFP-100B, TFP-100G | Pin No. FP-100A | I/O | Name and Function |
|---|---|---|---|---|---|
| Interrupts | NMI | 63 | 65 | Input | Nonmaskable interrupt: Requests a nonmaskable interrupt. When this pin is not used, it should be fixed high. |
| | $\overline{IRQ7}$ to $\overline{IRQ0}$ | 94, 93, 13, 12, 73 to 76 | 96, 95, 15, 14, 75 to 78 | Input | Interrupt request 7 to 0: These pins request a maskable interrupt. |
| Address bus | $A_{23}$ to $A_0$ | 2, 1, 100, 99, 53 to 50, 48 to 41, 39 to 32 | 4 to 1, 55 to 52, 50 to 43, 41 to 34 | Output | Address bus: These pins output an address. |
| Data bus | $D_{15}$ to $D_0$ | 30 to 19, 17 to 14 | 32 to 21, 19 to 16 | I/O | Data bus: These pins constitute a bidirectional data bus. |
| Bus control | $\overline{CS3}$ to $\overline{CS0}$ | 94 to 97 | 96 to 99 | Output | Chip select: Signals for selecting areas 3 to 0. |
| | $\overline{AS}$ | 70 | 72 | Output | Address strobe: When this pin is low, it indicates that address output on the address bus is enabled. |
| | $\overline{RD}$ | 71 | 73 | Output | Read: When this pin is low, it indicates that the external address space can be read. |
| | $\overline{HWR}$ | 72 | 74 | Output | High write: A strobe signal that writes to external space and indicates that the upper half ($D_{15}$ to $D_8$) of the data bus is enabled. |
| | $\overline{LWR}$ | 73 | 75 | Output | Low write: A strobe signal that writes to external space and indicates that the lower half ($D_7$ to $D_0$) of the data bus is enabled. |
| | $\overline{WAIT}$ | 74 | 76 | Input | Wait: Requests insertion of a wait state in the bus cycle when accessing external 3-state address space. |

**Table 1.3 Pin Functions (cont)**

| Type | Symbol | Pin No. FP-100B, TFP-100B, TFP-100G | Pin No. FP-100A | I/O | Name and Function |
|---|---|---|---|---|---|
| 16-bit timer-pulse unit (TPU) | TCLKD to TCLKA | 6, 4, 2, 1 | 8, 6, 4, 3 | Input | Clock input D to A: These pins input an external clock. |
| | TIOCA0, TIOCB0, TIOCC0, TIOCD0 | 99, 100, 1, 2 | 1 to 4 | I/O | Input capture/ output compare match A0 to D0: The TGR0A to TGR0D input capture input or output compare output, or PWM output pins. |
| | TIOCA1, TIOCB1 | 3, 4 | 5, 6 | I/O | Input capture/ output compare match A1 and B1: The TGR1A and TGR1B input capture input or output compare output, or PWM output pins. |
| | TIOCA2, TIOCB2 | 5, 6 | 7, 8 | I/O | Input capture/ output compare match A2 and B2: The TGR2A and TGR2B input capture input or output compare output, or PWM output pins. |
| | TIOCA3, TIOCB3, TIOCC3, TIOCD3 | 54 to 56, 59 | 56 to 58, 61 | I/O | Input capture/ output compare match A3 to D3: The TGR3A to TGR3D input capture input or output compare output, or PWM output pins. |
| | TIOCA4, TIOCB4 | 89, 90 | 91, 92 | I/O | Input capture/ output compare match A4 and B4: The TGR4A and TGR4B input capture input or output compare output, or PWM output pins. |
| | TIOCA5, TIOCB5 | 91, 92 | 93, 94 | I/O | Input capture/ output compare match A5 and B5: The TGR5A and TGR5B input capture input or output compare output, or PWM output pins. |
| 8-bit timer | TMO0, TMO1 | 91, 92 | 93, 94 | Output | Compare match output: The compare match output pins. |
| | TMCI0, TMCI1 | 59, 90 | 61, 92 | Input | Counter external clock input: Input pins for the external clock input to the counter. |
| | TMRI0, TMRI1 | 56, 89 | 58, 91 | Input | Counter external reset input: The counter reset input pins. |
| Watchdog timer (WDT) | $\overline{\text{WDTOVF}}$*2 | 60 | 62 | Output | Watchdog timer overflows: The counter overflows signal output pin in watchdog timer mode. |

**Table 1.3 Pin Functions (cont)**

| Type | Symbol | Pin No. FP-100B, TFP-100B, TFP-100G | Pin No. FP-100A | I/O | Name and Function |
|---|---|---|---|---|---|
| Serial communication interface (SCI) Smart Card interface | TxD1, TxD0 | 9, 8 | 11, 10 | Output | Transmit data (channel 0, 1): Data output pins. |
| | RxD1, RxD0 | 11, 10 | 13, 12 | Input | Receive data (channel 0, 1): Data input pins. |
| | SCK1 SCK0 | 13, 12 | 15, 14 | I/O | Serial clock (channel 0, 1): Clock I/O pins. |
| A/D converter | AN7 to AN0 | 86 to 79 | 88 to 81 | Input | Analog 7 to 0: Analog input pins. |
| | $\overline{\text{ADTRG}}$ | 93 | 95 | Input | A/D conversion external trigger input: Pin for input of an external trigger to start A/D conversion. |
| D/A converter | DA1, DA0 | 86, 85 | 88, 87 | Output | Analog output: D/A converter analog output pins. |
| A/D converter and D/A converters | $AV_{CC}$ | 77 | 79 | Input | This is the power supply pin for the A/D converter and D/A converter. When the A/D converter and D/A converter are not used, this pin should be connected to the system power supply (+5 V). |
| | $AV_{SS}$ | 87 | 89 | Input | This is the ground pin for the A/D converter and D/A converter. This pin should be connected to the system power supply (0 V). |
| | $V_{ref}$ | 78 | 80 | Input | This is the reference voltage input pin for the A/D converter and D/A converter. When the A/D converter and D/A converter are not used, this pin should be connected to the system power supply (+5 V). |
| I/O ports | $P1_7$ to $P1_0$ | 6 to 1, 100, 99 | 8 to 1 | I/O | Port 1: An 8-bit I/O port. Input or output can be designated for each bit by means of the port 1 data direction register (P1DDR). |
| | $P2_7$ to $P2_0$ | 92 to 89, 59, 56 to 54 | 94 to 91, 61, 58 to 56 | I/O | Port 2: An 8-bit I/O port. Input or output can be designated for each bit by means of the port 2 data direction register (P2DDR). |

**Table 1.3 Pin Functions (cont)**

| Type | Symbol | Pin No. FP-100B, TFP-100B, TFP-100G | Pin No. FP-100A | I/O | Name and Function |
|---|---|---|---|---|---|
| I/O ports | $P3_5$ to $P3_0$ | 13 to 8 | 15 to 10 | I/O | Port 3: A 6-bit I/O port. Input or output can be designated for each bit by means of the port 3 data direction register (P3DDR). |
| | $P4_7$ to $P4_0$ | 86 to 79 | 88 to 81 | Input | Port 4: An 8-bit input port. |
| | $PA_3$ to $PA_0$ | 53 to 50 | 55 to 52 | I/O | Port A: An 4-bit I/O port. Input or output can be designated for each bit by means of the port A data direction register (PADDR). |
| | $PB_7$ to $PB_0$ | 48 to 41 | 50 to 43 | I/O | Port B: An 8-bit I/O port. Input or output can be designated for each bit by means of the port B data direction register (PBDDR). |
| | $PC_7$ to $PC_0$ | 39 to 32 | 41 to 34 | I/O | Port C: An 8-bit I/O port. Input or output can be designated for each bit by means of the port C data direction register (PCDDR). |
| | $PD_7$ to $PD_0$ | 30 to 23 | 32 to 25 | I/O | Port D: An 8-bit I/O port. Input or output can be designated for each bit by means of the port D data direction register (PDDDR). |
| | $PE_7$ to $PE_0$ | 22 to 19, 17 to 14 | 24 to 21, 19 to 16 | I/O | Port E: An 8-bit I/O port. Input or output can be designated for each bit by means of the port E data direction register (PEDDR). |
| | $PF_7$ to $PF_0$ | 69 to 76 | 71 to 78 | I/O | Port F: An 8-bit I/O port. Input or output can be designated for each bit by means of the port F data direction register (PFDDR). |
| | $PG_4$ to $PG_0$ | 97 to 93 | 99 to 95 | I/O | Port G: A 5-bit I/O port. Input or output can be designated for each bit by means of the port G data direction register (PGDDR). |

Notes: 1. F-ZTAT version only.
2. Applies to ZTAT, mask ROM, and ROMless versions only.

# Section 2 CPU

## 2.1 Overview

The H8S/2000 CPU is a high-speed central processing unit with an internal 32-bit architecture that is upward-compatible with the H8/300 and H8/300H CPUs. The H8S/2000 CPU has sixteen 16-bit general registers, can address a 16-Mbyte (architecturally 4-Gbyte) linear address space, and is ideal for realtime control.

### 2.1.1 Features

The H8S/2000 CPU has the following features.

- Upward-compatible with H8/300 and H8/300H CPUs
  - Can execute H8/300 and H8/300H object programs
- General-register architecture
  - Sixteen 16-bit general registers (also usable as sixteen 8-bit registers or eight 32-bit registers)
- Sixty-five basic instructions
  - 8/16/32-bit arithmetic and logic instructions
  - Multiply and divide instructions
  - Powerful bit-manipulation instructions
- Eight addressing modes
  - Register direct [Rn]
  - Register indirect [@ERn]
  - Register indirect with displacement [@(d:16,ERn) or @(d:32,ERn)]
  - Register indirect with post-increment or pre-decrement [@ERn+ or @–ERn]
  - Absolute address [@aa:8, @aa:16, @aa:24, or @aa:32]
  - Immediate [#xx:8, #xx:16, or #xx:32]
  - Program-counter relative [@(d:8,PC) or @(d:16,PC)]
  - Memory indirect [@@aa:8]
- 16-Mbyte address space
  - Program: 16 Mbytes
  - Data: 16 Mbytes (4 Gbytes architecturally)

- High-speed operation
  - All frequently-used instructions execute in one or two states
  - Maximum clock rate : 20 MHz
  - 8/16/32-bit register-register add/subtract : 50 ns
  - 8 × 8-bit register-register multiply : 600 ns
  - 16 ÷ 8-bit register-register divide : 600 ns
  - 16 × 16-bit register-register multiply : 1000 ns
  - 32 ÷ 16-bit register-register divide : 1000 ns

- Two CPU operating modes
  - Normal mode (Supported on ZTAT, mask ROM, and ROMless versions only)
  - Advanced mode

- Power-down state
  - Transition to power-down state by SLEEP instruction
  - CPU clock speed selection

### 2.1.2 Differences between H8S/2600 CPU and H8S/2000 CPU

The differences between the H8S/2600 CPU and the H8S/2000 CPU are as shown below.

- Register configuration
  The MAC register is supported only by the H8S/2600 CPU.

- Basic instructions
  The four instructions MAC, CLRMAC, LDMAC, and STMAC are supported only by the H8S/2600 CPU.

- Number of execution states
  The number of execution states of the MULXU and MULXS instructions.

| | | Internal Operation | |
|---|---|---|---|
| **Instruction** | **Mnemonic** | **H8S/2600** | **H8S/2000** |
| MULXU | MULXU.B Rs, Rd | 3 | 12 |
| | MULXU.W Rs, ERd | 4 | 20 |
| MULXS | MULXS.B Rs, Rd | 4 | 13 |
| | MULXS.W Rs, ERd | 5 | 21 |

There are also differences in the address space, CCR and EXR register functions, power-down state, etc., depending on the product.

### 2.1.3 Differences from H8/300 CPU

In comparison to the H8/300 CPU, the H8S/2000 CPU has the following enhancements.

- More general registers and control registers
  - Eight 16-bit expanded registers, and one 8-bit control register, have been added.
- Expanded address space
  - Normal mode supports the same 64-kbyte address space as the H8/300 CPU. (ZTAT, mask ROM, and ROMless versions only)
  - Advanced mode supports a maximum 16-Mbyte address space.
- Enhanced addressing
  - The addressing modes have been enhanced to make effective use of the 16-Mbyte address space.
- Enhanced instructions
  - Addressing modes of bit-manipulation instructions have been enhanced.
  - Signed multiply and divide instructions have been added.
  - Two-bit shift instructions have been added.
  - Instructions for saving and restoring multiple registers have been added.
  - A test and set instruction has been added.
- Higher speed
  - Basic instructions execute twice as fast.

### 2.1.4 Differences from H8/300H CPU

In comparison to the H8/300H CPU, the H8S/2000 CPU has the following enhancements.

- Additional control register
  - One 8-bit control register has been added.
- Enhanced instructions
  - Addressing modes of bit-manipulation instructions have been enhanced.
  - Two-bit shift instructions have been added.
  - Instructions for saving and restoring multiple registers have been added.
  - A test and set instruction has been added.
- Higher speed
  - Basic instructions execute twice as fast.

## 2.2 CPU Operating Modes

The H8S/2000 CPU has two operating modes: normal and advanced. Normal mode supports a maximum 64-kbyte address space. Advanced mode supports a maximum 16-Mbyte total address space (architecturally a maximum 16-Mbyte program area and a maximum of 4 Gbytes for program and data areas combined). The mode is selected by the mode pins of the microcontroller.

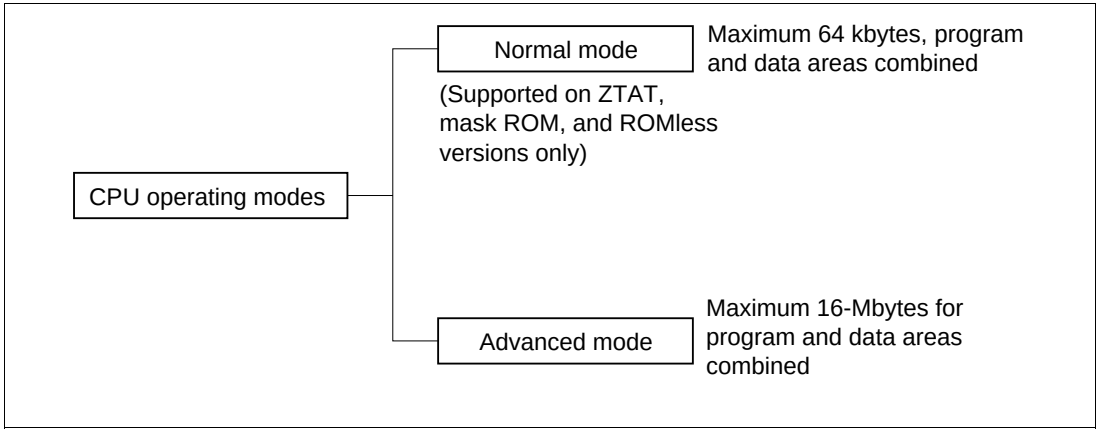



**Figure 2.1 CPU Operating Modes**

### (1) Normal Mode (ZTAT, Mask ROM, and ROMless Versions Only)

The exception vector table and stack have the same structure as in the H8/300 CPU.

**Address Space:** A maximum address space of 64 kbytes can be accessed.

**Extended Registers (En):** The extended registers (E0 to E7) can be used as 16-bit registers, or as the upper 16-bit segments of 32-bit registers. When En is used as a 16-bit register it can contain any value, even when the corresponding general register (Rn) is used as an address register. If the general register is referenced in the register indirect addressing mode with pre-decrement (@–Rn) or post-increment (@Rn+) and a carry or borrow occurs, however, the value in the corresponding extended register (En) will be affected.

**Instruction Set:** All instructions and addressing modes can be used. Only the lower 16 bits of effective addresses (EA) are valid.

**Exception Vector Table and Memory Indirect Branch Addresses:** In normal mode the top area starting at H'0000 is allocated to the exception vector table. One branch address is stored per 16 bits. The configuration of the exception vector table in normal mode is shown in figure 2.2. For details of the exception vector table, see section 4, Exception Handling.

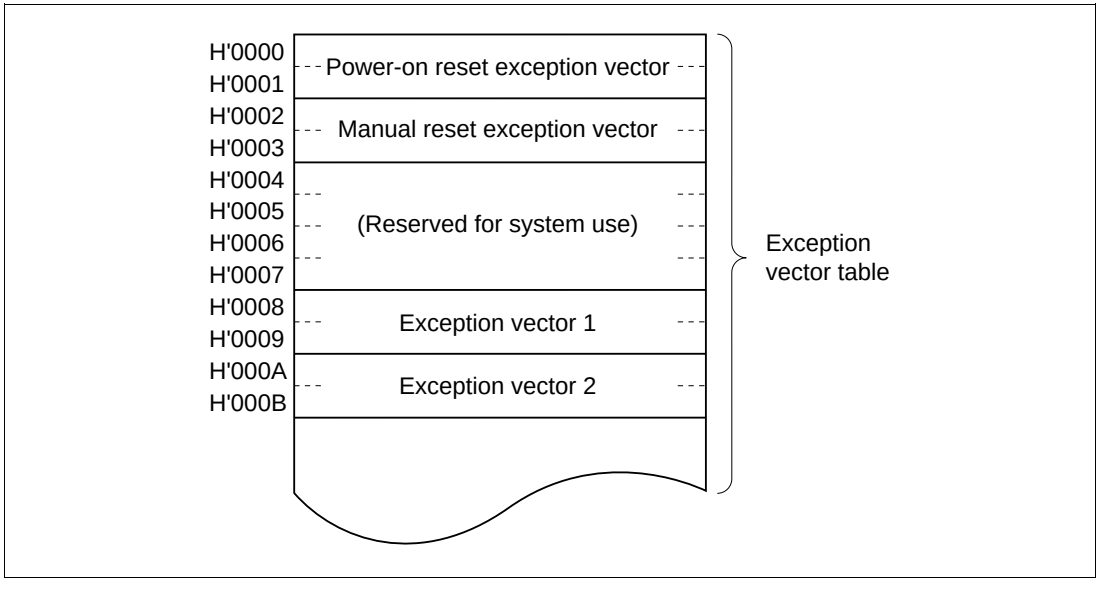
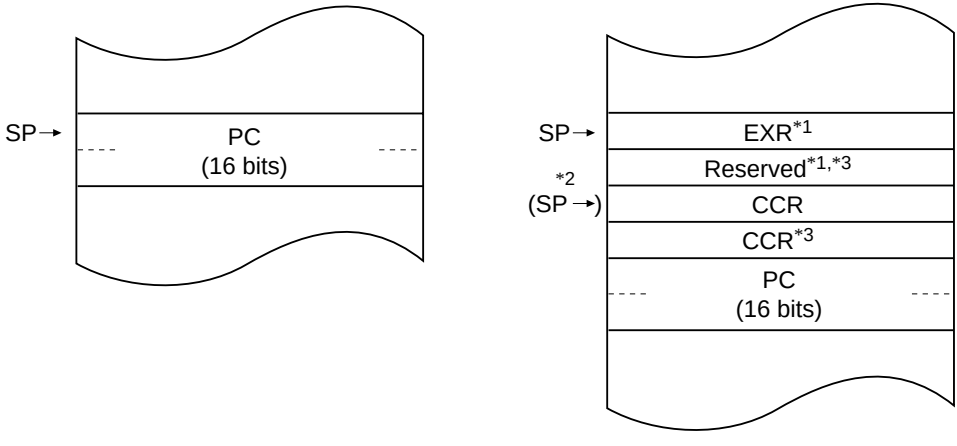



**Figure 2.2 Exception Vector Table (Normal Mode)**

The memory indirect addressing mode (@@aa:8) employed in the JMP and JSR instructions uses an 8-bit absolute address included in the instruction code to specify a memory operand that contains a branch address. In normal mode the operand is a 16-bit word operand, providing a 16-bit branch address. Branch addresses can be stored in the top area from H'0000 to H'00FF. Note that this area is also used for the exception vector table.

**Stack Structure:** When the program counter (PC) is pushed onto the stack in a subroutine call, and the PC, condition-code register (CCR), and extended control register (EXR) are pushed onto the stack in exception handling, they are stored as shown in figure 2.3. When EXR is invalid, it is not pushed onto the stack. For details, see section 4, Exception Handling.




**(a) Subroutine Branch** **(b) Exception Handling**

Notes: 1. When EXR is not used it is not stored on the stack.
2. SP when EXR is not used.
3. Ignored when returning.

**Figure 2.3 Stack Structure in Normal Mode**

**(2) Advanced Mode**

**Address Space:** Linear access is provided to a 16-Mbyte maximum address space (architecturally a maximum 16-Mbyte program area and a maximum 4-Gbyte data area, with a maximum of 4 Gbytes for program and data areas combined).

**Extended Registers (En):** The extended registers (E0 to E7) can be used as 16-bit registers, or as the upper 16-bit segments of 32-bit registers or address registers.

**Instruction Set:** All instructions and addressing modes can be used.

**Exception Vector Table and Memory Indirect Branch Addresses:** In advanced mode the top area starting at H'00000000 is allocated to the exception vector table in units of 32 bits. In each 32 bits, the upper 8 bits are ignored and a branch address is stored in the lower 24 bits (figure 2.4). For details of the exception vector table, see section 4, Exception Handling.

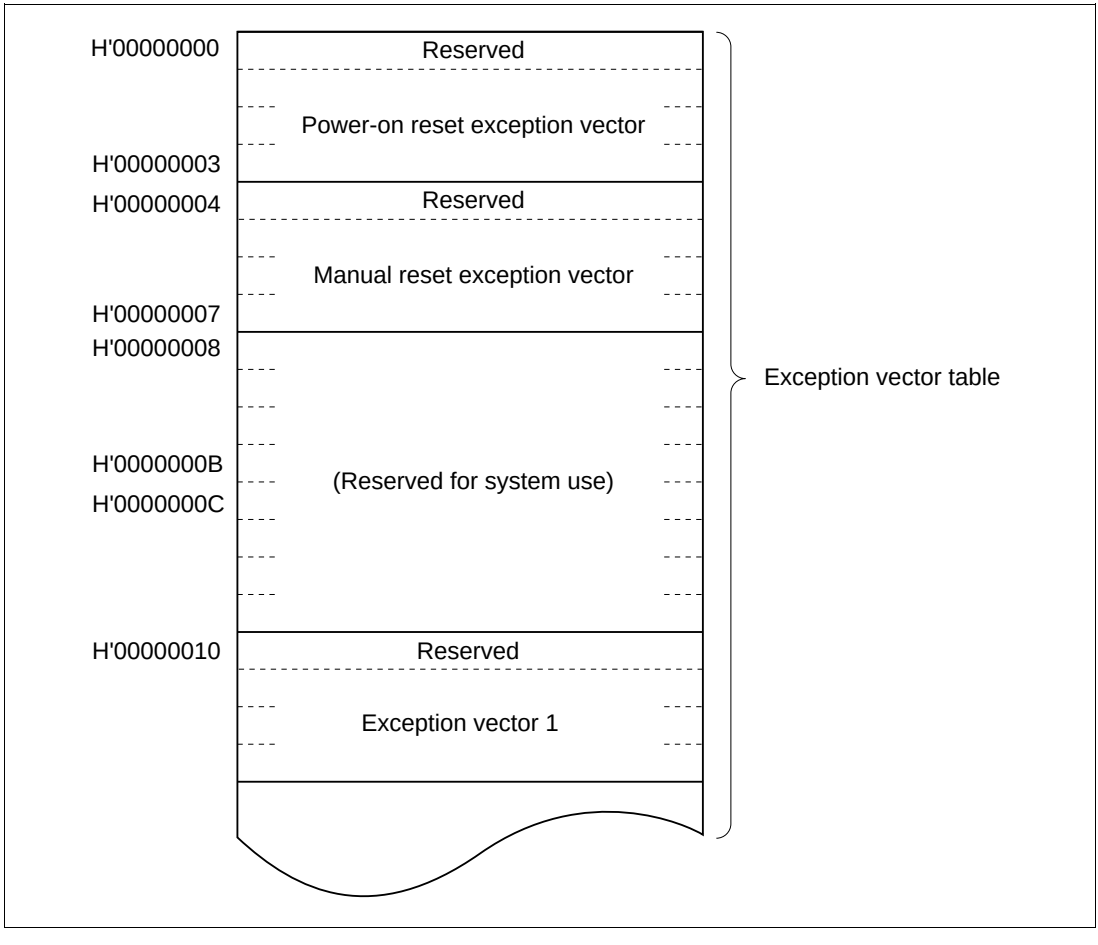
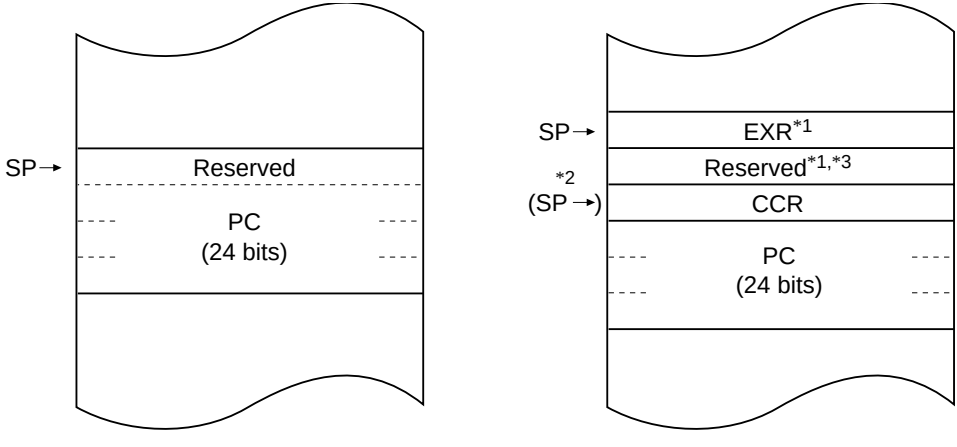



**Figure 2.4 Exception Vector Table (Advanced Mode)**

The memory indirect addressing mode (@@aa:8) employed in the JMP and JSR instructions uses an 8-bit absolute address included in the instruction code to specify a memory operand that contains a branch address. In advanced mode the operand is a 32-bit longword operand, providing a 32-bit branch address. The upper 8 bits of these 32 bits are a reserved area that is regarded as H'00. Branch addresses can be stored in the area from H'00000000 to H'000000FF. Note that the first part of this range is also the exception vector table.

**Stack Structure:** In advanced mode, when the program counter (PC) is pushed onto the stack in a subroutine call, and the PC, condition-code register (CCR), and extended control register (EXR) are pushed onto the stack in exception handling, they are stored as shown in figure 2.5. When EXR is invalid, it is not pushed onto the stack. For details, see section 4, Exception Handling.




**(a) Subroutine Branch** **(b) Exception Handling**

Notes: 1. When EXR is not used it is not stored on the stack.
2. SP when EXR is not used.
3. Ignored when returning.

**Figure 2.5 Stack Structure in Advanced Mode**

## 2.3 Address Space

Figure 2.6 shows a memory map of the H8S/2000 CPU. The H8S/2000 CPU provides linear access to a maximum 64-kbyte address space in normal mode*, and a maximum 16-Mbyte (architecturally 4-Gbyte) address space in advanced mode.

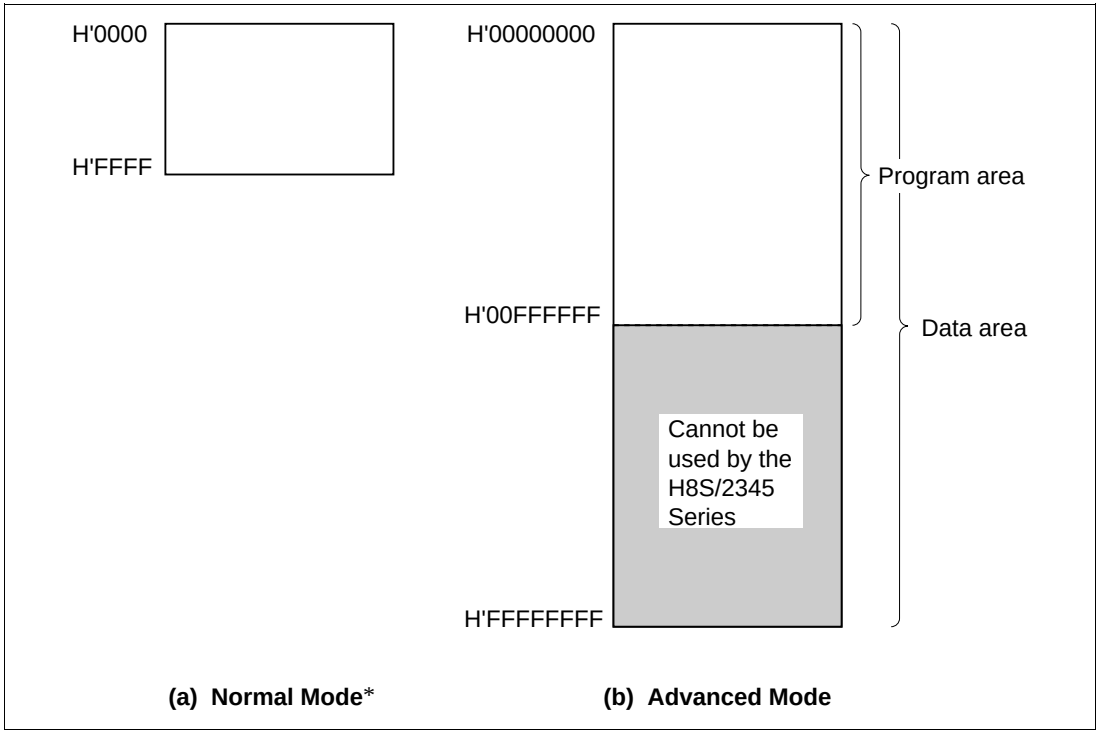



**Figure 2.6 Memory Map**

Note: * ZTAT, mask ROM, and ROMless versions only.

## 2.4 Register Configuration

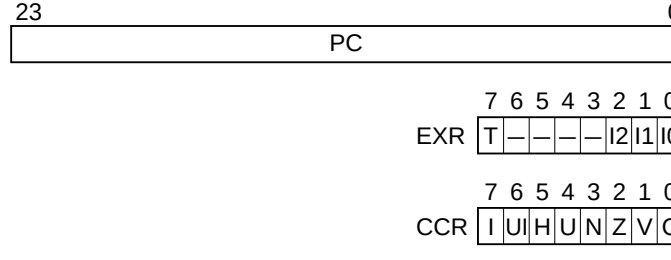
### 2.4.1 Overview

The CPU has the internal registers shown in figure 2.7. There are two types of registers: general registers and control registers.

**General Registers (Rn) and Extended Registers (En)**

| | 15 — 07 | 07 — 0 | |
|---|---|---|---|
| ER0 | E0 | R0H | R0L |
| ER1 | E1 | R1H | R1L |
| ER2 | E2 | R2H | R2L |
| ER3 | E3 | R3H | R3L |
| ER4 | E4 | R4H | R4L |
| ER5 | E5 | R5H | R5L |
| ER6 | E6 | R6H | R6L |
| ER7 (SP) | E7 | R7H | R7L |

**Control Registers (CR)**




**Legend**

| | | | |
|---|---|---|---|
| SP: | Stack pointer | H: | Half-carry flag |
| PC: | Program counter | U: | User bit |
| EXR: | Extended control register | N: | Negative flag |
| T: | Trace bit | Z: | Zero flag |
| I2 to I0: | Interrupt mask bits | V: | Overflow flag |
| CCR: | Condition-code register | C: | Carry flag |
| I: | Interrupt mask bit | | |
| UI: | User bit or interrupt mask bit* | | |

Note: * In the H8S/2345 Series, this bit cannot be used as an interrupt mask.

**Figure 2.7 CPU Registers**

### 2.4.2 General Registers

The CPU has eight 32-bit general registers. These general registers are all functionally alike and can be used as both address registers and data registers. When a general register is used as a data register, it can be accessed as a 32-bit, 16-bit, or 8-bit register. When the general registers are used as 32-bit registers or address registers, they are designated by the letters ER (ER0 to ER7).

The ER registers divide into 16-bit general registers designated by the letters E (E0 to E7) and R (R0 to R7). These registers are functionally equivalent, providing a maximum sixteen 16-bit registers. The E registers (E0 to E7) are also referred to as extended registers.

The R registers divide into 8-bit general registers designated by the letters RH (R0H to R7H) and RL (R0L to R7L). These registers are functionally equivalent, providing a maximum sixteen 8-bit registers.

Figure 2.8 illustrates the usage of the general registers. The usage of each register can be selected independently.

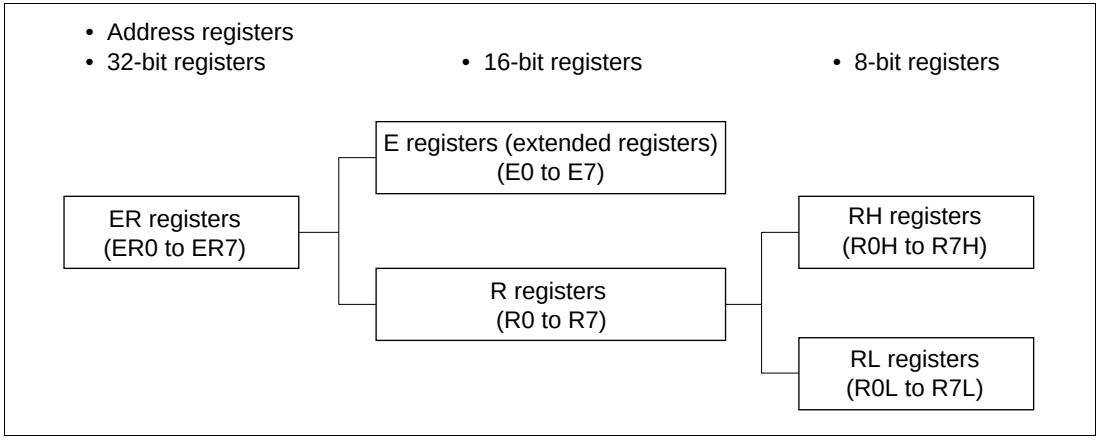



**Figure 2.8 Usage of General Registers**

General register ER7 has the function of stack pointer (SP) in addition to its general-register function, and is used implicitly in exception handling and subroutine calls. Figure 2.9 shows the stack.

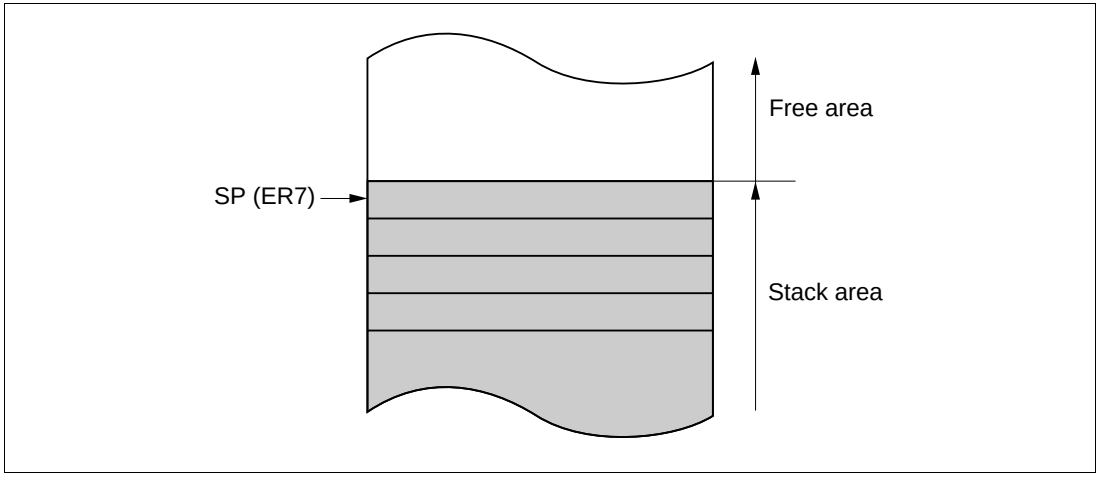



**Figure 2.9 Stack**

### 2.4.3 Control Registers

The control registers are the 24-bit program counter (PC), 8-bit extended control register (EXR), and 8-bit condition-code register (CCR).

**(1) Program Counter (PC):** This 24-bit counter indicates the address of the next instruction the CPU will execute. The length of all CPU instructions is 2 bytes (one word), so the least significant PC bit is ignored. (When an instruction is fetched, the least significant PC bit is regarded as 0.)

**(2) Extended Control Register (EXR):** This 8-bit register contains the trace bit (T) and three interrupt mask bits (I2 to I0).

**Bit 7—Trace Bit (T):** Selects trace mode. When this bit is cleared to 0, instructions are executed in sequence. When this bit is set to 1, a trace exception is generated each time an instruction is executed.

**Bits 6 to 3—Reserved:** These bits are reserved. They are always read as 1.

**Bits 2 to 0—Interrupt Mask Bits (I2 to I0):** These bits designate the interrupt mask level (0 to 7). For details, refer to section 5, Interrupt Controller.

Operations can be performed on the EXR bits by the LDC, STC, ANDC, ORC, and XORC instructions. All interrupts, including NMI, are disabled for three states after one of these instructions is executed, except for STC.

**(3) Condition-Code Register (CCR):** This 8-bit register contains internal CPU status information, including an interrupt mask bit (I) and half-carry (H), negative (N), zero (Z), overflow (V), and carry (C) flags.

**Bit 7—Interrupt Mask Bit (I):** Masks interrupts other than NMI when set to 1. (NMI is accepted regardless of the I bit setting.) The I bit is set to 1 by hardware at the start of an exception-handling sequence. For details, refer to section 5, Interrupt Controller.

**Bit 6—User Bit or Interrupt Mask Bit (UI):** Can be written and read by software using the LDC, STC, ANDC, ORC, and XORC instructions. With the H8S/2345 Series, this bit cannot be used as an interrupt mask bit.

**Bit 5—Half-Carry Flag (H):** When the ADD.B, ADDX.B, SUB.B, SUBX.B, CMP.B, or NEG.B instruction is executed, this flag is set to 1 if there is a carry or borrow at bit 3, and cleared to 0 otherwise. When the ADD.W, SUB.W, CMP.W, or NEG.W instruction is executed, the H flag is set to 1 if there is a carry or borrow at bit 11, and cleared to 0 otherwise. When the ADD.L, SUB.L, CMP.L, or NEG.L instruction is executed, the H flag is set to 1 if there is a carry or borrow at bit 27, and cleared to 0 otherwise.

**Bit 4—User Bit (U):** Can be written and read by software using the LDC, STC, ANDC, ORC, and XORC instructions.

**Bit 3—Negative Flag (N):** Stores the value of the most significant bit (sign bit) of data.

**Bit 2—Zero Flag (Z):** Set to 1 to indicate zero data, and cleared to 0 to indicate non-zero data.

**Bit 1—Overflow Flag (V):** Set to 1 when an arithmetic overflow occurs, and cleared to 0 at other times.

**Bit 0—Carry Flag (C):** Set to 1 when a carry occurs, and cleared to 0 otherwise. Used by:

- Add instructions, to indicate a carry
- Subtract instructions, to indicate a borrow
- Shift and rotate instructions, to store the value shifted out of the end bit

The carry flag is also used as a bit accumulator by bit manipulation instructions.

Some instructions leave some or all of the flag bits unchanged. For the action of each instruction on the flag bits, refer to Appendix A.1, List of Instructions.

Operations can be performed on the CCR bits by the LDC, STC, ANDC, ORC, and XORC instructions. The N, Z, V, and C flags are used as branching conditions for conditional branch (Bcc) instructions.

### 2.4.4 Initial Register Values

Reset exception handling loads the CPU's program counter (PC) from the vector table, clears the trace bit in EXR to 0, and sets the interrupt mask bits in CCR and EXR to 1. The other CCR bits and the general registers are not initialized. In particular, the stack pointer (ER7) is not initialized. The stack pointer should therefore be initialized by an MOV.L instruction executed immediately after a reset.

## 2.5 Data Formats

The CPU can process 1-bit, 4-bit (BCD), 8-bit (byte), 16-bit (word), and 32-bit (longword) data. Bit-manipulation instructions operate on 1-bit data by accessing bit n (n = 0, 1, 2, …, 7) of byte operand data. The DAA and DAS decimal-adjust instructions treat byte data as two digits of 4-bit BCD data.

### 2.5.1 General Register Data Formats

Figure 2.10 shows the data formats in general registers.

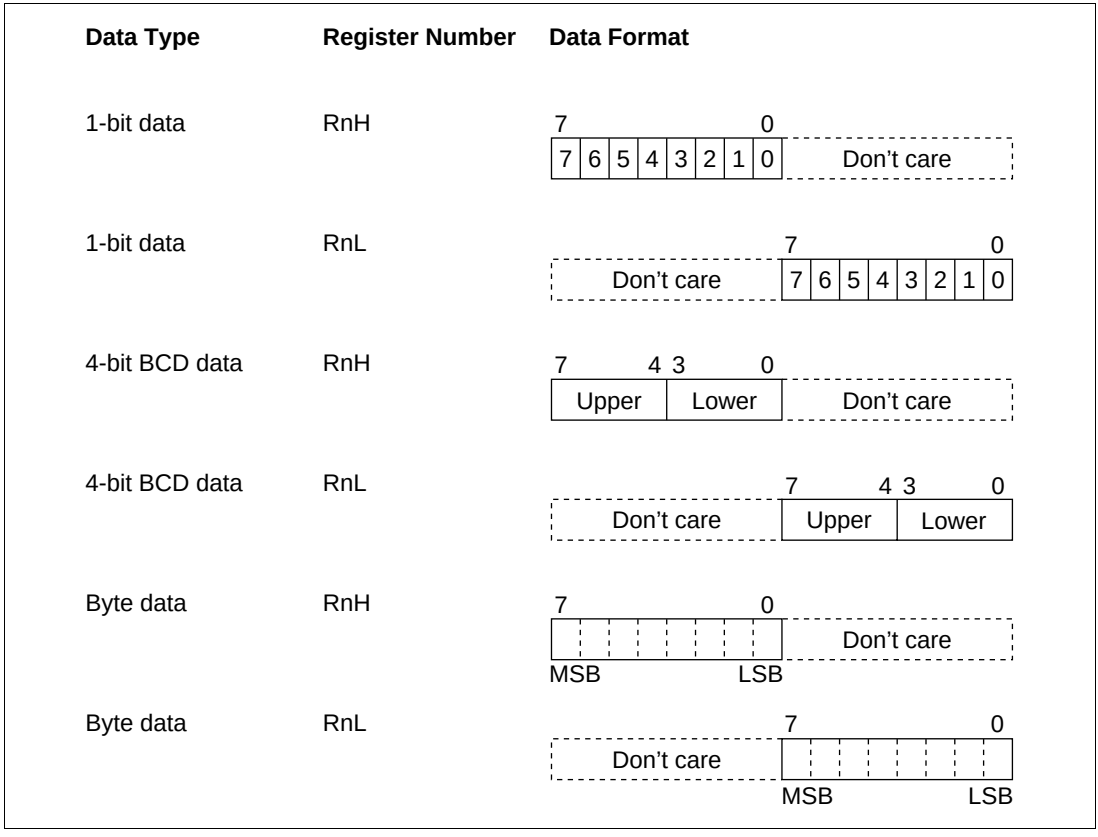



**Figure 2.10 General Register Data Formats**

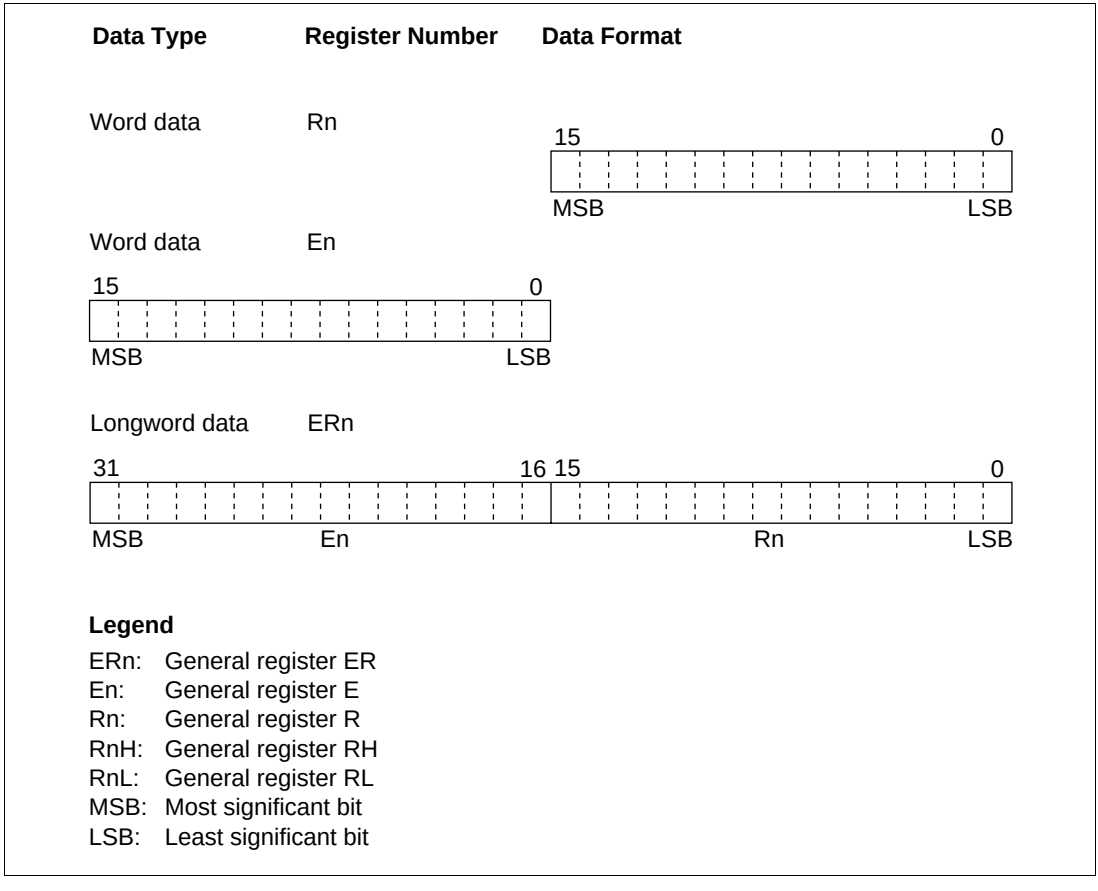



**Figure 2.10 General Register Data Formats (cont)**

### 2.5.2 Memory Data Formats

Figure 2.11 shows the data formats in memory. The CPU can access word data and longword data in memory, but word or longword data must begin at an even address. If an attempt is made to access word or longword data at an odd address, no address error occurs but the least significant bit of the address is regarded as 0, so the access starts at the preceding address. This also applies to instruction fetches.

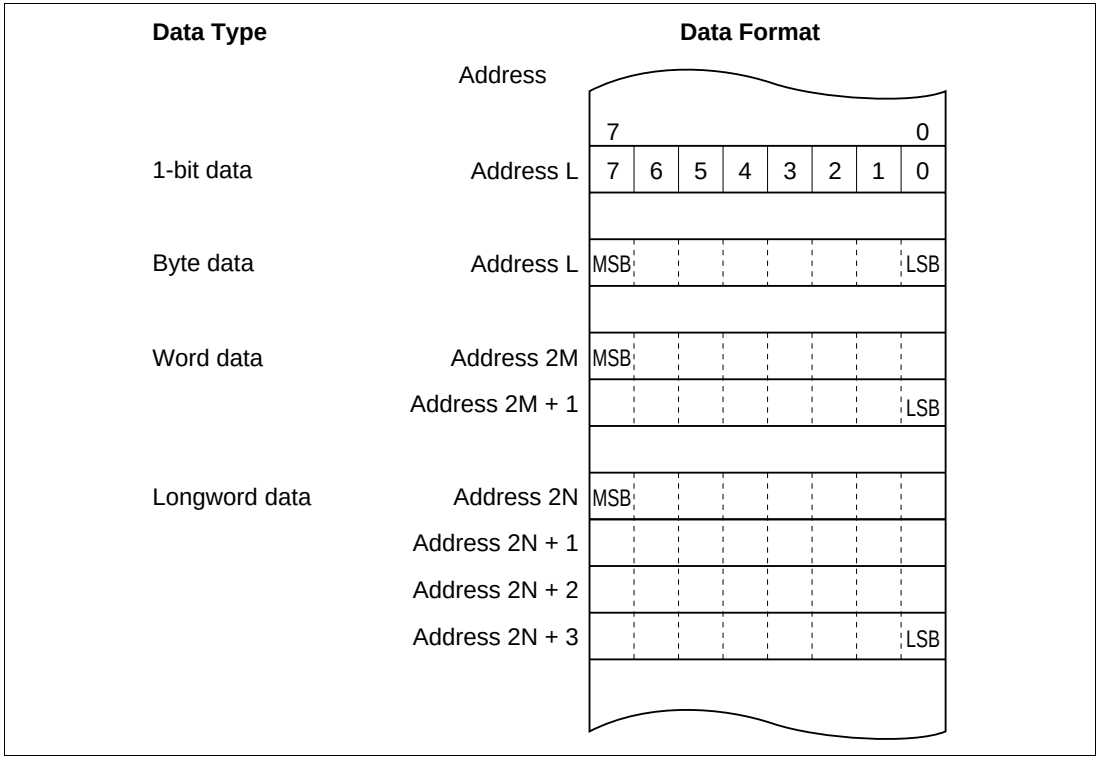



**Figure 2.11 Memory Data Formats**

When ER7 is used as an address register to access the stack, the operand size should be word size or longword size.

## 2.6 Instruction Set

### 2.6.1 Overview

The H8S/2000 CPU has 65 types of instructions. The instructions are classified by function in table 2.1.

**Table 2.1 Instruction Classification**

| Function | Instructions | Size | Types |
|---|---|---|---|
| Data transfer | MOV | BWL | 5 |
| | POP*[1], PUSH*[1] | WL | |
| | LDM, STM | L | |
| | MOVFPE, MOVTPE*[3] | B | |
| Arithmetic operations | ADD, SUB, CMP, NEG | BWL | 19 |
| | ADDX, SUBX, DAA, DAS | B | |
| | INC, DEC | BWL | |
| | ADDS, SUBS | L | |
| | MULXU, DIVXU, MULXS, DIVXS | BW | |
| | EXTU, EXTS | WL | |
| | TAS | B | |
| Logic operations | AND, OR, XOR, NOT | BWL | 4 |
| Shift | SHAL, SHAR, SHLL, SHLR, ROTL, ROTR, ROTXL, ROTXR | BWL | 8 |
| Bit manipulation | BSET, BCLR, BNOT, BTST, BLD, BILD, BST, BIST, BAND, BIAND, BOR, BIOR, BXOR, BIXOR | B | 14 |
| Branch | Bcc*[2], JMP, BSR, JSR, RTS | — | 5 |
| System control | TRAPA, RTE, SLEEP, LDC, STC, ANDC, ORC, XORC, NOP | — | 9 |
| Block data transfer | EEPMOV | — | 1 |

Notes: B-byte size; W-word size; L-longword size.

1. POP.W Rn and PUSH.W Rn are identical to MOV.W @SP+, Rn and MOV.W Rn, @-SP. POP.L ERn and PUSH.L ERn are identical to MOV.L @SP+, ERn and MOV.L ERn, @-SP.
2. Bcc is the general name for conditional branch instructions.
3. Cannot be used in the H8S/2345 Series.

### 2.6.2 Instructions and Addressing Modes

Table 2.2 indicates the combinations of instructions and addressing modes that the H8S/2600 CPU can use.

**Table 2.2 Combinations of Instructions and Addressing Modes**

| Function | Instruction | Addressing Modes | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | #xx | Rn | @ERn | @(d:16,ERn) | @(d:32,ERn) | @–ERn/@ERn+ | @aa:8 | @aa:16 | @aa:24 | @aa:32 | @(d:8,PC) | @(d:16,PC) | @@aa:8 | \| |
| Data transfer | MOV | BWL | BWL | BWL | BWL | BWL | BWL | B | BWL | — | BWL | — | — | — | — |
| | POP, PUSH | — | — | — | — | — | — | — | — | — | — | — | — | — | WL |
| | LDM, STM | — | — | — | — | — | — | — | — | — | — | — | — | — | L |
| | MOVFPE*, MOVTPE* | — | — | — | — | — | — | — | B | — | — | — | — | — | — |
| Arithmetic operations | ADD, CMP | BWL | BWL | — | — | — | — | — | — | — | — | — | — | — | — |
| | SUB | WL | BWL | — | — | — | — | — | — | — | — | — | — | — | — |
| | ADDX, SUBX | B | B | — | — | — | — | — | — | — | — | — | — | — | — |
| | ADDS, SUBS | — | L | — | — | — | — | — | — | — | — | — | — | — | — |
| | INC, DEC | — | BWL | — | — | — | — | — | — | — | — | — | — | — | — |
| | DAA, DAS | — | B | — | — | — | — | — | — | — | — | — | — | — | — |
| | MULXU, DIVXU | — | BW | — | — | — | — | — | — | — | — | — | — | — | — |
| | MULXS, DIVXS | — | BW | — | — | — | — | — | — | — | — | — | — | — | — |
| | NEG | — | BWL | — | — | — | — | — | — | — | — | — | — | — | — |
| | EXTU, EXTS | — | WL | — | — | — | — | — | — | — | — | — | — | — | — |
| | TAS | — | — | B | — | — | — | — | — | — | — | — | — | — | — |
| Logic operations | AND, OR, XOR | BWL | BWL | — | — | — | — | — | — | — | — | — | — | — | — |
| | NOT | — | BWL | — | — | — | — | — | — | — | — | — | — | — | — |
| Shift | | — | BWL | — | — | — | — | — | — | — | — | — | — | — | — |
| Bit manipulation | | — | B | B | — | — | — | B | B | — | B | — | — | — | — |
| Branch | Bcc, BSR | — | — | — | — | — | — | — | — | — | — | ◯ | ◯ | — | — |
| | JMP, JSR | — | — | — | — | — | — | — | — | ◯ | — | — | — | ◯ | — |
| | RTS | — | — | — | — | — | — | — | — | — | — | — | — | — | ◯ |

Note: * Cannot be used in the H8S/2345 Series.

**Table 2.2 Combinations of Instructions and Addressing Modes (cont)**

| Function | Instruction | Addressing Modes | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | #xx | Rn | @ERn | @(d:16,ERn) | @(d:32,ERn) | @–ERn/@ERn+ | @aa:8 | @aa:16 | @aa:24 | @aa:32 | @(d:8,PC) | @(d:16,PC) | @@aa:8 | — |
| System control | TRAPA | — | — | — | — | — | — | — | — | — | — | — | — | — | ○ |
| | RTE | — | — | — | — | — | — | — | — | — | — | — | — | — | ○ |
| | SLEEP | — | — | — | — | — | — | — | — | — | — | — | — | — | ○ |
| | LDC | B | B | W | W | W | W | — | W | — | W | — | — | — | — |
| | STC | — | B | W | W | W | W | — | W | — | W | — | — | — | — |
| | ANDC, ORC, XORC | B | — | — | — | — | — | — | — | — | — | — | — | — | — |
| | NOP | — | — | — | — | — | — | — | — | — | — | — | — | — | ○ |
| Block data transfer | | — | — | — | — | — | — | — | — | — | — | — | — | — | BW |

Legend:

B: Byte

W: Word

L: Longword

### 2.6.3 Table of Instructions Classified by Function

Table 2.3 summarizes the instructions in each functional category. The notation used in table 2.3 is defined below.

**Operation Notation**

| | |
|---|---|
| Rd | General register (destination)* |
| Rs | General register (source)* |
| Rn | General register* |
| ERn | General register (32-bit register) |
| (EAd) | Destination operand |
| (EAs) | Source operand |
| EXR | Extended control register |
| CCR | Condition-code register |
| N | N (negative) flag in CCR |
| Z | Z (zero) flag in CCR |
| V | V (overflow) flag in CCR |
| C | C (carry) flag in CCR |
| PC | Program counter |
| SP | Stack pointer |
| #IMM | Immediate data |
| disp | Displacement |
| + | Addition |
| – | Subtraction |
| × | Multiplication |
| ÷ | Division |
| ∧ | Logical AND |
| ∨ | Logical OR |
| ⊕ | Logical exclusive OR |
| → | Move |
| ¬ | NOT (logical complement) |
| :8/:16/:24/:32 | 8-, 16-, 24-, or 32-bit length |

Note: * General registers include 8-bit registers (R0H to R7H, R0L to R7L), 16-bit registers (R0 to R7, E0 to E7), and 32-bit registers (ER0 to ER7).

**Table 2.3 Instructions Classified by Function**

| Type | Instruction | Size* | Function |
|---|---|---|---|
| Data transfer | MOV | B/W/L | (EAs) → Rd, Rs → (Ead)<br>Moves data between two general registers or between a general register and memory, or moves immediate data to a general register. |
| | MOVFPE | B | Cannot be used in the H8S/2345 Series. |
| | MOVTPE | B | Cannot be used in the H8S/2345 Series. |
| | POP | W/L | @SP+ → Rn<br>Pops a register from the stack. POP.W Rn is identical to MOV.W @SP+, Rn. POP.L ERn is identical to MOV.L @SP+, ERn. |
| | PUSH | W/L | Rn → @–SP<br>Pushes a register onto the stack. PUSH.W Rn is identical to MOV.W Rn, @–SP. PUSH.L ERn is identical to MOV.L ERn, @–SP. |
| | LDM | L | @SP+ → Rn (register list)<br>Pops two or more general registers from the stack. |
| | STM | L | Rn (register list) → @–SP<br>Pushes two or more general registers onto the stack. |

Note: * Size refers to the operand size.
B: Byte
W: Word
L: Longword

**Table 2.3 Instructions Classified by Function (cont)**

| Type | Instruction | Size* | Function |
|---|---|---|---|
| Arithmetic operations | ADD<br>SUB | B/W/L | Rd ± Rs → Rd, Rd ± #IMM → Rd<br>Performs addition or subtraction on data in two general registers, or on immediate data and data in a general register. (Immediate byte data cannot be subtracted from byte data in a general register. Use the SUBX or ADD instruction.) |
| | ADDX<br>SUBX | B | Rd ± Rs ± C → Rd, Rd ± #IMM ± C → Rd<br>Performs addition or subtraction with carry or borrow on byte data in two general registers, or on immediate data and data in a general register. |
| | INC<br>DEC | B/W/L | Rd ± 1 → Rd, Rd ± 2 → Rd<br>Increments or decrements a general register by 1 or 2. (Byte operands can be incremented or decremented by 1 only.) |
| | ADDS<br>SUBS | L | Rd ± 1 → Rd, Rd ± 2 → Rd, Rd ± 4 → Rd<br>Adds or subtracts the value 1, 2, or 4 to or from data in a 32-bit register. |
| | DAA<br>DAS | B | Rd decimal adjust → Rd<br>Decimal-adjusts an addition or subtraction result in a general register by referring to the CCR to produce 4-bit BCD data. |
| | MULXU | B/W | Rd × Rs → Rd<br>Performs unsigned multiplication on data in two general registers: either 8 bits × 8 bits → 16 bits or 16 bits × 16 bits → 32 bits. |
| | MULXS | B/W | Rd × Rs → Rd<br>Performs signed multiplication on data in two general registers: either 8 bits × 8 bits → 16 bits or 16 bits × 16 bits → 32 bits. |
| | DIVXU | B/W | Rd ÷ Rs → Rd<br>Performs unsigned division on data in two general registers: either 16 bits ÷ 8 bits → 8-bit quotient and 8-bit remainder or 32 bits ÷ 16 bits → 16-bit quotient and 16-bit remainder. |

Note: * Size refers to the operand size.
B: Byte
W: Word
L: Longword

**Table 2.3 Instructions Classified by Function (cont)**

| Type | Instruction | Size* | Function |
|---|---|---|---|
| Arithmetic operations | DIVXS | B/W | Rd ÷ Rs → Rd<br>Performs signed division on data in two general registers: either 16 bits ÷ 8 bits → 8-bit quotient and 8-bit remainder or 32 bits ÷ 16 bits → 16-bit quotient and 16-bit remainder. |
| | CMP | B/W/L | Rd – Rs, Rd – #IMM<br>Compares data in a general register with data in another general register or with immediate data, and sets CCR bits according to the result. |
| | NEG | B/W/L | 0 – Rd → Rd<br>Takes the two's complement (arithmetic complement) of data in a general register. |
| | EXTU | W/L | Rd (zero extension) → Rd<br>Extends the lower 8 bits of a 16-bit register to word size, or the lower 16 bits of a 32-bit register to longword size, by padding with zeros on the left. |
| | EXTS | W/L | Rd (sign extension) → Rd<br>Extends the lower 8 bits of a 16-bit register to word size, or the lower 16 bits of a 32-bit register to longword size, by extending the sign bit. |
| | TAS | B | @ERd – 0, 1 → (<bit 7> of @Erd)<br>Tests memory contents, and sets the most significant bit (bit 7) to 1. |

Note: * Size refers to the operand size.
B: Byte
W: Word
L: Longword

**Table 2.3 Instructions Classified by Function (cont)**

| Type | Instruction | Size* | Function |
|---|---|---|---|
| Logic operations | AND | B/W/L | Rd $\wedge$ Rs $\rightarrow$ Rd, Rd $\wedge$ #IMM $\rightarrow$ Rd<br>Performs a logical AND operation on a general register and another general register or immediate data. |
| | OR | B/W/L | Rd $\vee$ Rs $\rightarrow$ Rd, Rd $\vee$ #IMM $\rightarrow$ Rd<br>Performs a logical OR operation on a general register and another general register or immediate data. |
| | XOR | B/W/L | Rd $\oplus$ Rs $\rightarrow$ Rd, Rd $\oplus$ #IMM $\rightarrow$ Rd<br>Performs a logical exclusive OR operation on a general register and another general register or immediate data. |
| | NOT | B/W/L | $\neg$ (Rd) $\rightarrow$ (Rd)<br>Takes the one's complement of general register contents. |
| Shift operations | SHAL<br>SHAR | B/W/L | Rd (shift) $\rightarrow$ Rd<br>Performs an arithmetic shift on general register contents.<br>1-bit or 2-bit shift is possible. |
| | SHLL<br>SHLR | B/W/L | Rd (shift) $\rightarrow$ Rd<br>Performs a logical shift on general register contents.<br>1-bit or 2-bit shift is possible. |
| | ROTL<br>ROTR | B/W/L | Rd (rotate) $\rightarrow$ Rd<br>Rotates general register contents.<br>1-bit or 2-bit rotation is possible. |
| | ROTXL<br>ROTXR | B/W/L | Rd (rotate) $\rightarrow$ Rd<br>Rotates general register contents through the carry flag.<br>1-bit or 2-bit rotation is possible. |

Note: * Size refers to the operand size.

B: Byte

W: Word

L: Longword

**Table 2.3 Instructions Classified by Function (cont)**

| Type | Instruction | Size* | Function |
|---|---|---|---|
| Bit-manipulation instructions | BSET | B | 1 → (<bit-No.> of <EAd>)<br>Sets a specified bit in a general register or memory operand to 1. The bit number is specified by 3-bit immediate data or the lower three bits of a general register. |
| | BCLR | B | 0 → (<bit-No.> of <EAd>)<br>Clears a specified bit in a general register or memory operand to 0. The bit number is specified by 3-bit immediate data or the lower three bits of a general register. |
| | BNOT | B | ¬ (<bit-No.> of <EAd>) → (<bit-No.> of <EAd>)<br>Inverts a specified bit in a general register or memory operand. The bit number is specified by 3-bit immediate data or the lower three bits of a general register. |
| | BTST | B | ¬ (<bit-No.> of <EAd>) → Z<br>Tests a specified bit in a general register or memory operand and sets or clears the Z flag accordingly. The bit number is specified by 3-bit immediate data or the lower three bits of a general register. |
| | BAND | B | C ∧ (<bit-No.> of <EAd>) → C<br>ANDs the carry flag with a specified bit in a general register or memory operand and stores the result in the carry flag. |
| | BIAND | B | C ∧ ¬ (<bit-No.> of <EAd>) → C<br>ANDs the carry flag with the inverse of a specified bit in a general register or memory operand and stores the result in the carry flag.<br>The bit number is specified by 3-bit immediate data. |
| | BOR | B | C ∨ (<bit-No.> of <EAd>) → C<br>ORs the carry flag with a specified bit in a general register or memory operand and stores the result in the carry flag. |
| | BIOR | B | C ∨ ¬ (<bit-No.> of <EAd>) → C<br>ORs the carry flag with the inverse of a specified bit in a general register or memory operand and stores the result in the carry flag.<br>The bit number is specified by 3-bit immediate data. |

Note: * Size refers to the operand size.
B: Byte

**Table 2.3 Instructions Classified by Function (cont)**

| Type | Instruction | Size* | Function |
|---|---|---|---|
| Bit-manipulation instructions | BXOR | B | C ⊕ (<bit-No.> of <EAd>) → C<br>Exclusive-ORs the carry flag with a specified bit in a general register or memory operand and stores the result in the carry flag. |
| | BIXOR | B | C ⊕ ¬ (<bit-No.> of <EAd>) → C<br>Exclusive-ORs the carry flag with the inverse of a specified bit in a general register or memory operand and stores the result in the carry flag.<br>The bit number is specified by 3-bit immediate data. |
| | BLD | B | (<bit-No.> of <EAd>) → C<br>Transfers a specified bit in a general register or memory operand to the carry flag. |
| | BILD | B | ¬ (<bit-No.> of <EAd>) → C<br>Transfers the inverse of a specified bit in a general register or memory operand to the carry flag.<br>The bit number is specified by 3-bit immediate data. |
| | BST | B | C → (<bit-No.> of <EAd>)<br>Transfers the carry flag value to a specified bit in a general register or memory operand. |
| | BIST | B | ¬ C → (<bit-No.> of <EAd>)<br>Transfers the inverse of the carry flag value to a specified bit in a general register or memory operand.<br>The bit number is specified by 3-bit immediate data. |

Note: * Size refers to the operand size.
B: Byte

**Table 2.3 Instructions Classified by Function (cont)**

| Type | Instruction | Size* | Function |
|---|---|---|---|
| Branch instructions | Bcc | — | Branches to a specified address if a specified condition is true. The branching conditions are listed below. |

| Mnemonic | Description | Condition |
|---|---|---|
| BRA(BT) | Always (true) | Always |
| BRN(BF) | Never (false) | Never |
| BHI | High | $C \vee Z = 0$ |
| BLS | Low or same | $C \vee Z = 1$ |
| BCC(BHS) | Carry clear (high or same) | $C = 0$ |
| BCS(BLO) | Carry set (low) | $C = 1$ |
| BNE | Not equal | $Z = 0$ |
| BEQ | Equal | $Z = 1$ |
| BVC | Overflow clear | $V = 0$ |
| BVS | Overflow set | $V = 1$ |
| BPL | Plus | $N = 0$ |
| BMI | Minus | $N = 1$ |
| BGE | Greater or equal | $N \oplus V = 0$ |
| BLT | Less than | $N \oplus V = 1$ |
| BGT | Greater than | $Z \vee (N \oplus V) = 0$ |
| BLE | Less or equal | $Z \vee (N \oplus V) = 1$ |

| | | | |
|---|---|---|---|
| | JMP | — | Branches unconditionally to a specified address. |
| | BSR | — | Branches to a subroutine at a specified address. |
| | JSR | — | Branches to a subroutine at a specified address. |
| | RTS | — | Returns from a subroutine |

**Table 2.3 Instructions Classified by Function (cont)**

| Type | Instruction | Size* | Function |
|---|---|---|---|
| System control instructions | TRAPA | — | Starts trap-instruction exception handling. |
| | RTE | — | Returns from an exception-handling routine. |
| | SLEEP | — | Causes a transition to a power-down state. |
| | LDC | B/W | (EAs) → CCR, (EAs) → EXR<br>Moves the source operand contents or immediate data to CCR or EXR. Although CCR and EXR are 8-bit registers, word-size transfers are performed between them and memory. The upper 8 bits are valid. |
| | STC | B/W | CCR → (EAd), EXR → (EAd)<br>Transfers CCR or EXR contents to a general register or memory. Although CCR and EXR are 8-bit registers, word-size transfers are performed between them and memory. The upper 8 bits are valid. |
| | ANDC | B | CCR ∧ #IMM → CCR, EXR ∧ #IMM → EXR<br>Logically ANDs the CCR or EXR contents with immediate data. |
| | ORC | B | CCR ∨ #IMM → CCR, EXR ∨ #IMM → EXR<br>Logically ORs the CCR or EXR contents with immediate data. |
| | XORC | B | CCR ⊕ #IMM → CCR, EXR ⊕ #IMM → EXR<br>Logically exclusive-ORs the CCR or EXR contents with immediate data. |
| | NOP | — | PC + 2 → PC<br>Only increments the program counter. |

Note: * Size refers to the operand size.
B: Byte
W: Word

**Table 2.3 Instructions Classified by Function (cont)**

| Type | Instruction | Size | Function |
|---|---|---|---|
| Block data transfer instruction | EEPMOV.B | — | if R4L ≠ 0 then<br>Repeat @ER5+ → @ER6+<br>R4L–1 → R4L<br>Until R4L = 0<br>else next; |
| | EEPMOV.W | — | if R4 ≠ 0 then<br>Repeat @ER5+ → @ER6+<br>R4–1 → R4<br>Until R4 = 0<br>else next; |
| | | | Transfers a data block according to parameters set in general registers R4L or R4, ER5, and ER6. |
| | | | R4L or R4: size of block (bytes)<br>ER5: starting source address<br>ER6: starting destination address |
| | | | Execution of the next instruction begins as soon as the transfer is completed. |

### 2.6.4 Basic Instruction Formats

The CPU instructions consist of 2-byte (1-word) units. An instruction consists of an operation field (op field), a register field (r field), an effective address extension (EA field), and a condition field (cc).

Figure 2.12 shows examples of instruction formats.

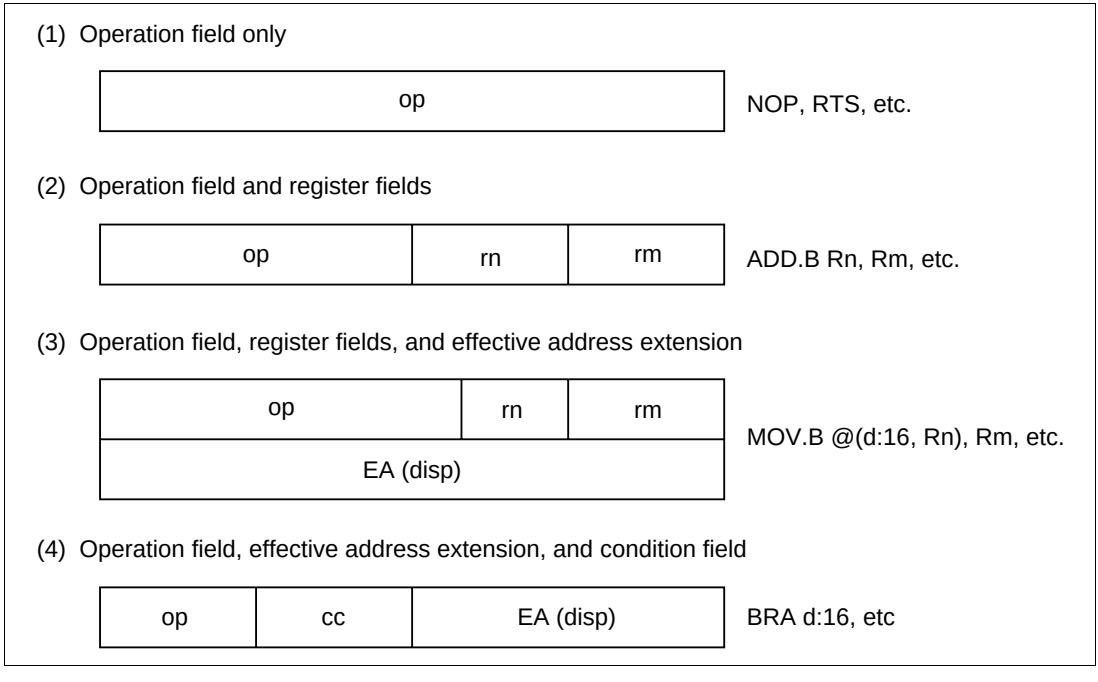



**Figure 2.12 Instruction Formats (Examples)**

**(1) Operation Field:** Indicates the function of the instruction, the addressing mode, and the operation to be carried out on the operand. The operation field always includes the first four bits of the instruction. Some instructions have two operation fields.

**(2) Register Field:** Specifies a general register. Address registers are specified by 3 bits, data registers by 3 bits or 4 bits. Some instructions have two register fields. Some have no register field.

**(3) Effective Address Extension:** Eight, 16, or 32 bits specifying immediate data, an absolute address, or a displacement.

**(4) Condition Field:** Specifies the branching condition of Bcc instructions.

## 2.7 Addressing Modes and Effective Address Calculation

### 2.7.1 Addressing Mode

The CPU supports the eight addressing modes listed in table 2.4. Each instruction uses a subset of these addressing modes. Arithmetic and logic instructions can use the register direct and immediate modes. Data transfer instructions can use all addressing modes except program-counter relative and memory indirect. Bit manipulation instructions use register direct, register indirect, or absolute addressing mode to specify an operand, and register direct (BSET, BCLR, BNOT, and BTST instructions) or immediate (3-bit) addressing mode to specify a bit number in the operand.

**Table 2.4 Addressing Modes**

| No. | Addressing Mode | Symbol |
|---|---|---|
| 1 | Register direct | Rn |
| 2 | Register indirect | @ERn |
| 3 | Register indirect with displacement | @(d:16,ERn)/@(d:32,ERn) |
| 4 | Register indirect with post-increment<br>Register indirect with pre-decrement | @ERn+<br>@–ERn |
| 5 | Absolute address | @aa:8/@aa:16/@aa:24/@aa:32 |
| 6 | Immediate | #xx:8/#xx:16/#xx:32 |
| 7 | Program-counter relative | @(d:8,PC)/@(d:16,PC) |
| 8 | Memory indirect | @@aa:8 |

**(1) Register Direct—Rn:** The register field of the instruction specifies an 8-, 16-, or 32-bit general register containing the operand. R0H to R7H and R0L to R7L can be specified as 8-bit registers. R0 to R7 and E0 to E7 can be specified as 16-bit registers. ER0 to ER7 can be specified as 32-bit registers.

**(2) Register Indirect—@ERn:** The register field of the instruction code specifies an address register (ERn) which contains the address of the operand on memory. If the address is a program instruction address, the lower 24 bits are valid and the upper 8 bits are all assumed to be 0 (H'00).

**(3) Register Indirect with Displacement—@(d:16, ERn) or @(d:32, ERn):** A 16-bit or 32-bit displacement contained in the instruction is added to an address register (ERn) specified by the register field of the instruction, and the sum gives the address of a memory operand. A 16-bit displacement is sign-extended when added.

**(4) Register Indirect with Post-Increment or Pre-Decrement—@ERn+ or @-ERn:**

- Register indirect with post-increment—@ERn+

  The register field of the instruction code specifies an address register (ERn) which contains the address of a memory operand. After the operand is accessed, 1, 2, or 4 is added to the address register contents and the sum is stored in the address register. The value added is 1 for byte access, 2 for word transfer instruction, or 4 for longword transfer instruction. For word or longword transfer instruction, the register value should be even.
- Register indirect with pre-decrement—@-ERn

  The value 1, 2, or 4 is subtracted from an address register (ERn) specified by the register field in the instruction code, and the result becomes the address of a memory operand. The result is also stored in the address register. The value subtracted is 1 for byte access, 2 for word transfer instruction, or 4 for longword transfer instruction. For word or longword transfer instruction, the register value should be even.

**(5) Absolute Address—@aa:8, @aa:16, @aa:24, or @aa:32:** The instruction code contains the absolute address of a memory operand. The absolute address may be 8 bits long (@aa:8), 16 bits long (@aa:16), 24 bits long (@aa:24), or 32 bits long (@aa:32).

To access data, the absolute address should be 8 bits (@aa:8), 16 bits (@aa:16), or 32 bits (@aa:32) long. For an 8-bit absolute address, the upper 24 bits are all assumed to be 1 (H'FFFF). For a 16-bit absolute address the upper 16 bits are a sign extension. A 32-bit absolute address can access the entire address space.

A 24-bit absolute address (@aa:24) indicates the address of a program instruction. The upper 8 bits are all assumed to be 0 (H'00).

Table 2.5 indicates the accessible absolute address ranges.

**Table 2.5 Absolute Address Access Ranges**

| Absolute Address | | Normal Mode* | Advanced Mode |
|---|---|---|---|
| Data address | 8 bits (@aa:8) | H'FF00 to H'FFFF | H'FFFF00 to H'FFFFFF |
| | 16 bits (@aa:16) | H'0000 to H'FFFF | H'000000 to H'007FFF, H'FF8000 to H'FFFFFF |
| | 32 bits (@aa:32) | | H'000000 to H'FFFFFF |
| Program instruction address | 24 bits (@aa:24) | | |

Note: * ZTAT, mask ROM, and ROMless versions only.

**(6) Immediate—#xx:8, #xx:16, or #xx:32:** The instruction contains 8-bit (#xx:8), 16-bit (#xx:16), or 32-bit (#xx:32) immediate data as an operand.

The ADDS, SUBS, INC, and DEC instructions contain immediate data implicitly. Some bit manipulation instructions contain 3-bit immediate data in the instruction code, specifying a bit number. The TRAPA instruction contains 2-bit immediate data in its instruction code, specifying a vector address.

**(7) Program-Counter Relative—@(d:8, PC) or @(d:16, PC):** This mode is used in the Bcc and BSR instructions. An 8-bit or 16-bit displacement contained in the instruction is sign-extended and added to the 24-bit PC contents to generate a branch address. Only the lower 24 bits of this branch address are valid; the upper 8 bits are all assumed to be 0 (H'00). The PC value to which the displacement is added is the address of the first byte of the next instruction, so the possible branching range is –126 to +128 bytes (–63 to +64 words) or –32766 to +32768 bytes (–16383 to +16384 words) from the branch instruction. The resulting value should be an even number.

**(8) Memory Indirect—@@aa:8:** This mode can be used by the JMP and JSR instructions. The instruction code contains an 8-bit absolute address specifying a memory operand. This memory operand contains a branch address. The upper bits of the absolute address are all assumed to be 0, so the address range is 0 to 255 (H'0000 to H'00FF in normal mode, H'000000 to H'0000FF in advanced mode). In normal mode* the memory operand is a word operand and the branch address is 16 bits long. In advanced mode the memory operand is a longword operand, the first byte of which is assumed to be all 0 (H'00).

Note that the first part of the address range is also the exception vector area. For further details, refer to section 4, Exception Handling.

Note: * ZTAT, mask ROM, and ROMless versions only.

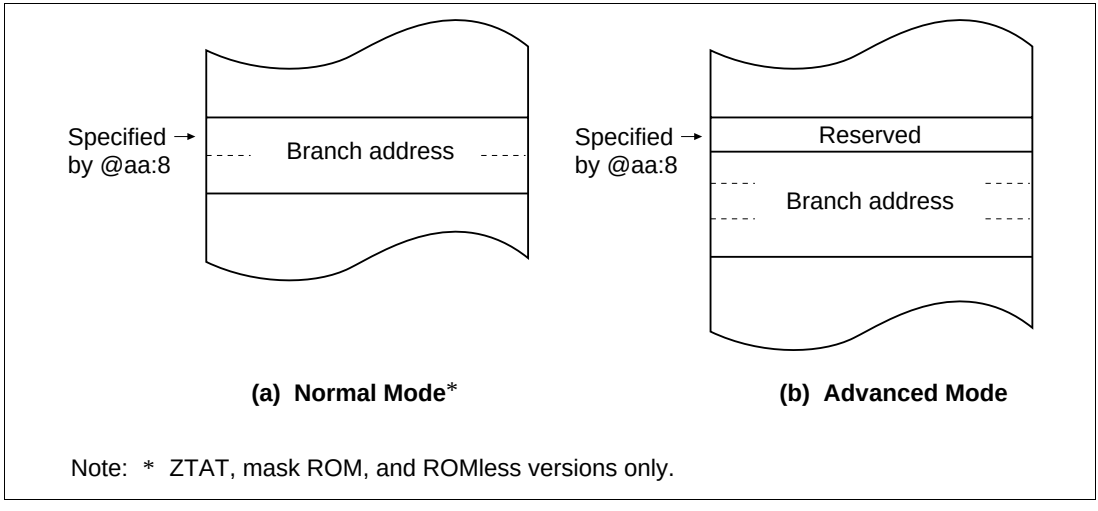



**Figure 2.13 Branch Address Specification in Memory Indirect Mode**

If an odd address is specified in word or longword memory access, or as a branch address, the least significant bit is regarded as 0, causing data to be accessed or instruction code to be fetched at the address preceding the specified address. (For further information, see section 2.5.2, Memory Data Formats.)

### 2.7.2 Effective Address Calculation

Table 2.6 indicates how effective addresses are calculated in each addressing mode. In normal mode* the upper 8 bits of the effective address are ignored in order to generate a 16-bit address.

Note: * ZTAT, mask ROM, and ROMless versions only.

**Table 2.6 Effective Address Calculation**

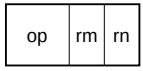
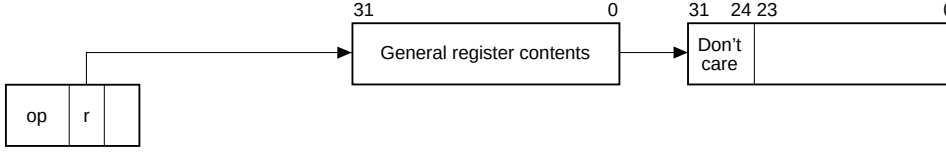
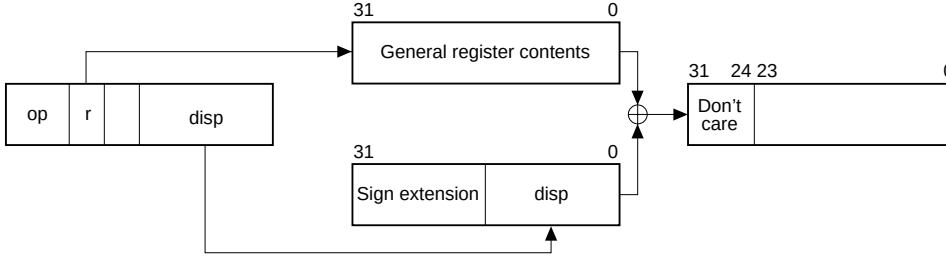
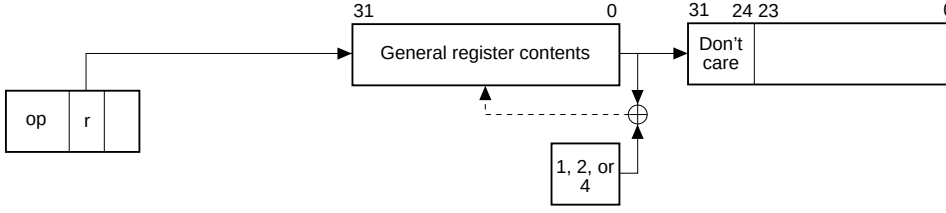
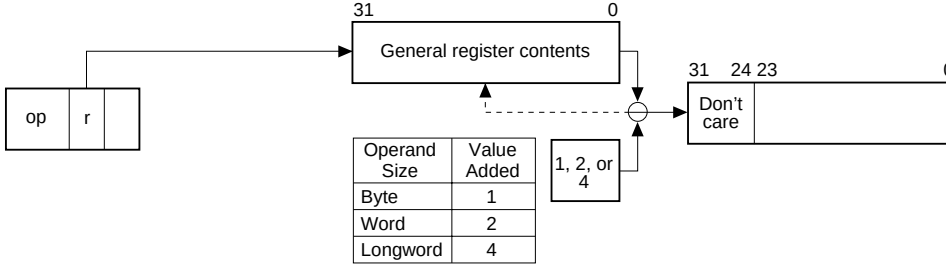
| No. | Addressing Mode and Instruction Format | Effective Address Calculation | Effective Address (EA) |
|---|---|---|---|
| 1 | Register direct (Rn)<br>  | | Operand is general register contents. |
| 2 | Register indirect (@ERn)<br>  | | |
| 3 | Register indirect with displacement<br>@(d:16, ERn) or @(d:32, ERn)<br>  | | |
| 4 | Register indirect with post-increment or pre-decrement<br>• Register indirect with post-increment @ERn+<br>  | | |
| | • Register indirect with pre-decrement @–ERn<br>  <br>Operand Size / Value Added<br>Byte 1<br>Word 2<br>Longword 4 | | |

**Table 2.6 Effective Address Calculation (cont)**

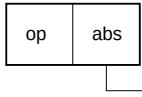
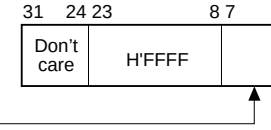
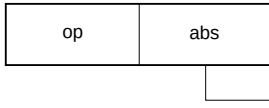
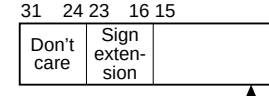
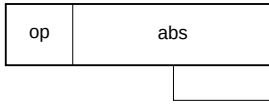
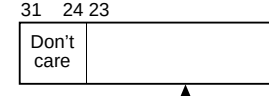
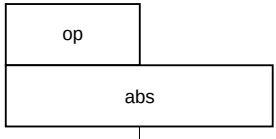
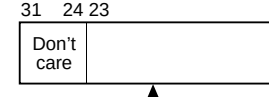
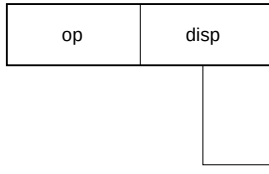
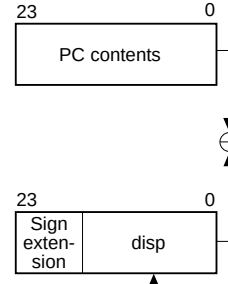
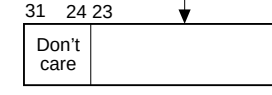
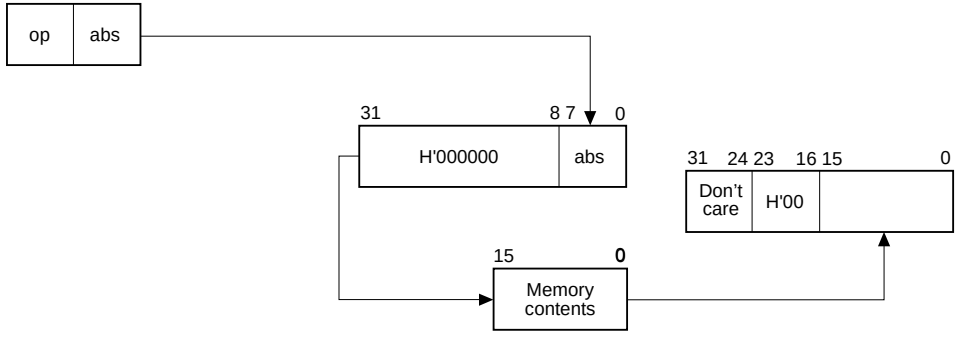
| No. | Addressing Mode and Instruction Format | Effective Address Calculation | Effective Address (EA) |
|---|---|---|---|
| 5 | Absolute address | | |
| | @aa:8 | | |
| |  | |  |
| | @aa:16 | | |
| |  | |  |
| | @aa:24 | | |
| |  | |  |
| | @aa:32 | | |
| |  | |  |
| 6 | Immediate #xx:8/#xx:16/#xx:32 | | Operand is immediate data. |
| | | | |
| 7 | Program-counter relative | | |
| | @(d:8, PC)/@(d:16, PC) | | |
| |  |  |  |

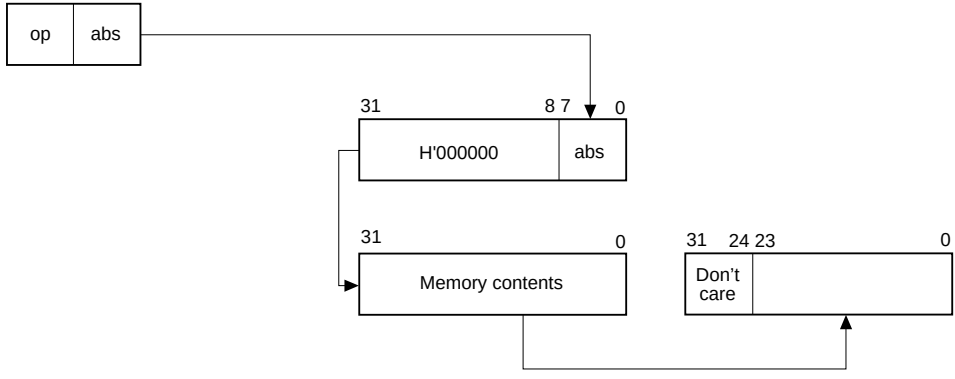
**Table 2.6 Effective Address Calculation (cont)**

| No. | Addressing Mode and Instruction Format | Effective Address Calculation | Effective Address (EA) |
|---|---|---|---|
| 8 | Memory indirect @@aa:8 | | |

- Normal mode*




- Advanced mode




Note: * ZTAT, mask ROM, and ROMless versions only.

## 2.8 Processing States

### 2.8.1 Overview

The CPU has five main processing states: the reset state, exception handling state, program execution state, bus-released state, and power-down state. Figure 2.14 shows a diagram of the processing states. Figure 2.15 indicates the state transitions.

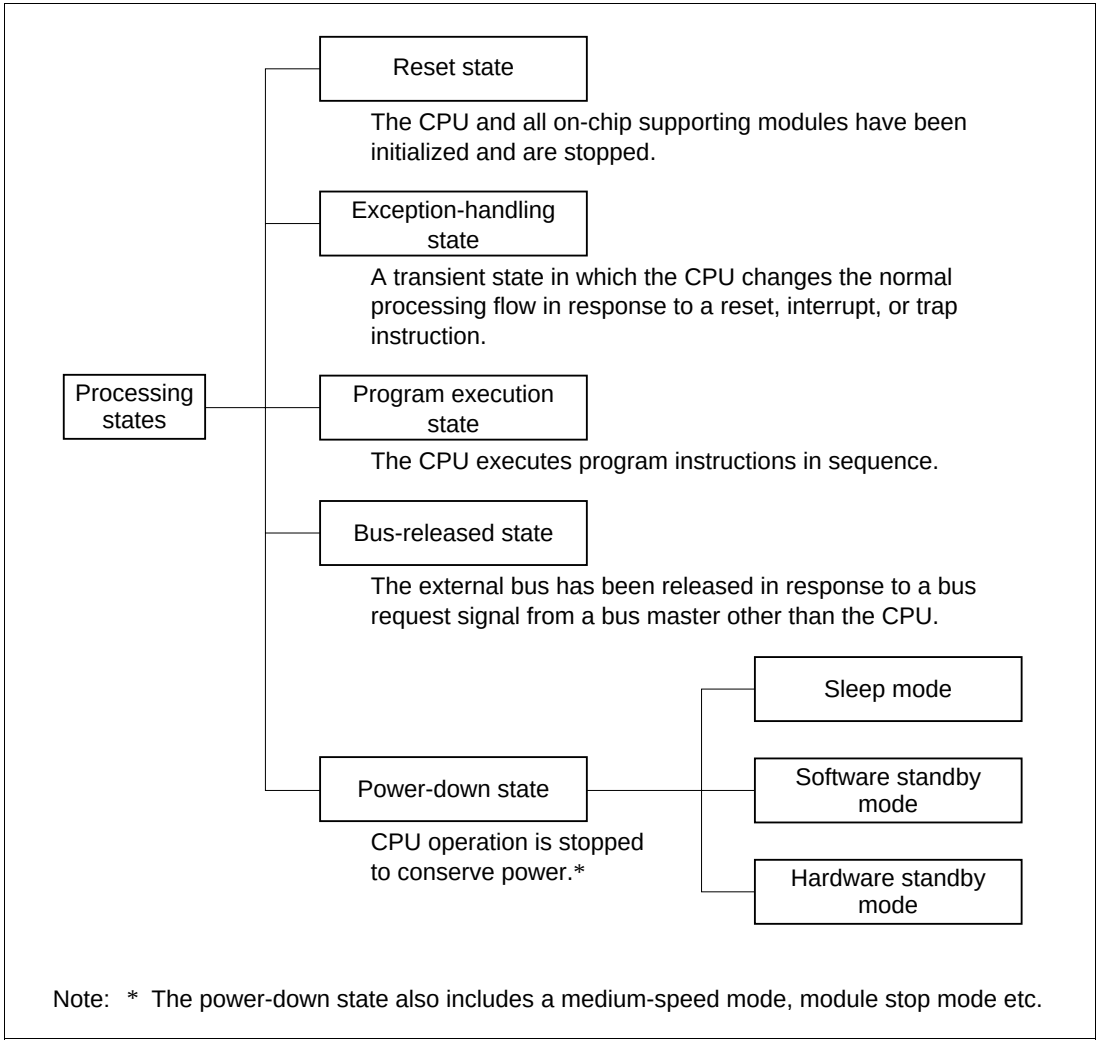
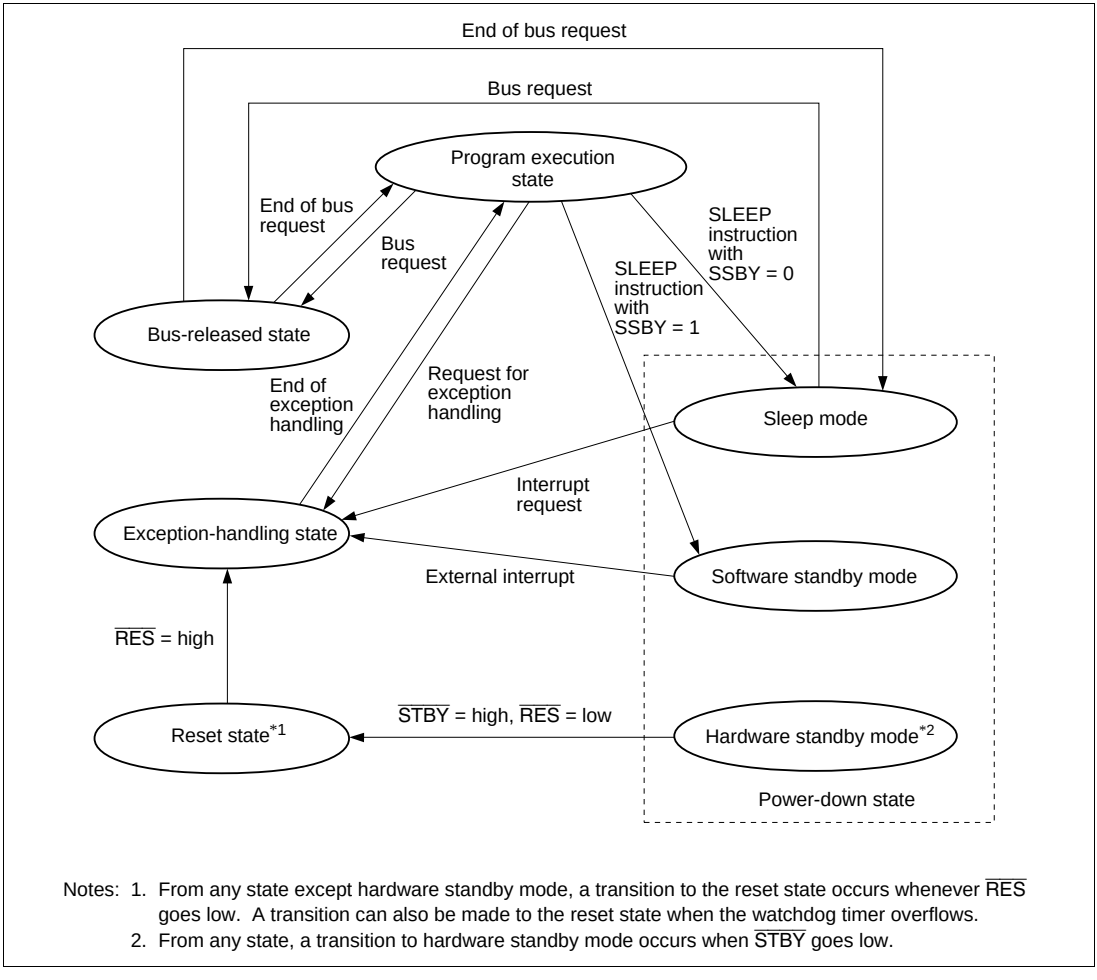



**Figure 2.14 Processing States**




Notes: 1. From any state except hardware standby mode, a transition to the reset state occurs whenever $\overline{RES}$ goes low. A transition can also be made to the reset state when the watchdog timer overflows.
2. From any state, a transition to hardware standby mode occurs when $\overline{STBY}$ goes low.

**Figure 2.15 State Transitions**

### 2.8.2 Reset State

When the $\overline{RES}$ input goes low all current processing stops and the CPU enters the reset state. The CPU enters the power-on reset state when the NMI pin is high, or the manual reset state when the NMI pin is low. All interrupts are masked in the reset state. Reset exception handling starts when the $\overline{RES}$ signal changes from low to high.

The reset state can also be entered by a watchdog timer overflow. For details, refer to section 11, Watchdog Timer.

### 2.8.3 Exception-Handling State

The exception-handling state is a transient state that occurs when the CPU alters the normal processing flow due to a reset, interrupt, or trap instruction. The CPU fetches a start address (vector) from the exception vector table and branches to that address.

#### (1) Types of Exception Handling and Their Priority

Exception handling is performed for traces, resets, interrupts, and trap instructions. Table 2.7 indicates the types of exception handling and their priority. Trap instruction exception handling is always accepted, in the program execution state.

Exception handling and the stack structure depend on the interrupt control mode set in SYSCR.

**Table 2.7 Exception Handling Types and Priority**

| Priority | Type of Exception | Detection Timing | Start of Exception Handling |
|---|---|---|---|
| High ↑ Low | Reset | Synchronized with clock | Exception handling starts immediately after a low-to-high transition at the $\overline{\text{RES}}$ pin, or when the watchdog timer overflows. |
| | Trace | End of instruction execution or end of exception-handling sequence*1 | When the trace (T) bit is set to 1, the trace starts at the end of the current instruction or current exception-handling sequence |
| | Interrupt | End of instruction execution or end of exception-handling sequence*2 | When an interrupt is requested, exception handling starts at the end of the current instruction or current exception-handling sequence |
| | Trap instruction | When TRAPA instruction is executed | Exception handling starts when a trap (TRAPA) instruction is executed*3 |

Notes: 1. Traces are enabled only in interrupt control mode 2. Trace exception-handling is not executed at the end of the RTE instruction.
2. Interrupts are not detected at the end of the ANDC, ORC, XORC, and LDC instructions, or immediately after reset exception handling.
3. Trap instruction exception handling is always accepted, in the program execution state.

**(2) Reset Exception Handling**

After the $\overline{\text{RES}}$ pin has gone low and the reset state has been entered, when $\overline{\text{RES}}$ goes high again, reset exception handling starts. The CPU enters the power-on reset state when the NMI pin is high, or the manual reset state when the NMI pin is low. When reset exception handling starts the CPU fetches a start address (vector) from the exception vector table and starts program execution from that address. All interrupts, including NMI, are disabled during reset exception handling and after it ends.

**(3) Traces**

Traces are enabled only in interrupt control mode 2. Trace mode is entered when the T bit of EXR is set to 1. When trace mode is established, trace exception handling starts at the end of each instruction.

At the end of a trace exception-handling sequence, the T bit of EXR is cleared to 0 and trace mode is cleared. Interrupt masks are not affected.

The T bit saved on the stack retains its value of 1, and when the RTE instruction is executed to return from the trace exception-handling routine, trace mode is entered again. Trace exception-handling is not executed at the end of the RTE instruction.

Trace mode is not entered in interrupt control mode 0, regardless of the state of the T bit.

**(4) Interrupt Exception Handling and Trap Instruction Exception Handling**

When interrupt or trap-instruction exception handling begins, the CPU references the stack pointer (ER7) and pushes the program counter and other control registers onto the stack. Next, the CPU alters the settings of the interrupt mask bits in the control registers. Then the CPU fetches a start address (vector) from the exception vector table and program execution starts from that start address.

Figure 2.16 shows the stack after exception handling ends.

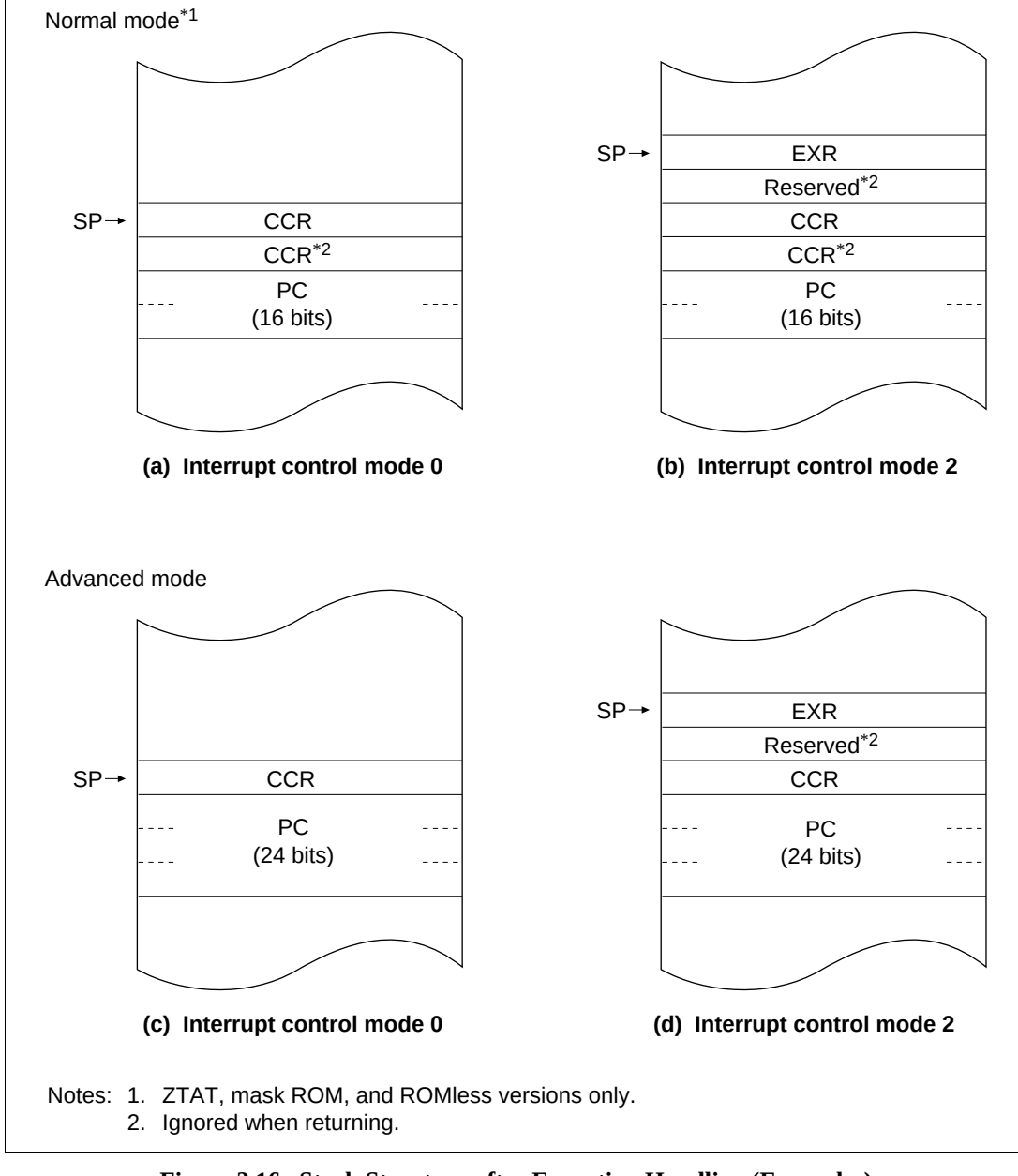



**Figure 2.16 Stack Structure after Exception Handling (Examples)**

### 2.8.4 Program Execution State

In this state the CPU executes program instructions in sequence.

### 2.8.5 Bus-Released State

This is a state in which the bus has been released in response to a bus request from a bus master other than the CPU. While the bus is released, the CPU halts operations.

There is one other bus master in addition to the CPU: the data transfer controller (DTC).

For further details, refer to section 6, Bus Controller.

### 2.8.6 Power-Down State

The power-down state includes both modes in which the CPU stops operating and modes in which the CPU does not stop. There are three modes in which the CPU stops operating: sleep mode, software standby mode, and hardware standby mode. There are also two other power-down modes: medium-speed mode, and module stop mode. In medium-speed mode the CPU and other bus masters operate on a medium-speed clock. Module stop mode permits halting of the operation of individual modules, other than the CPU. For details, refer to section 19, Power-Down State.

**(1) Sleep Mode:** A transition to sleep mode is made if the SLEEP instruction is executed while the software standby bit (SSBY) in the standby control register (SBYCR) is cleared to 0. In sleep mode, CPU operations stop immediately after execution of the SLEEP instruction. The contents of CPU registers are retained.

**(2) Software Standby Mode:** A transition to software standby mode is made if the SLEEP instruction is executed while the SSBY bit in SBYCR is set to 1. In software standby mode, the CPU and clock halt and all MCU operations stop. As long as a specified voltage is supplied, the contents of CPU registers and on-chip RAM are retained. The I/O ports also remain in their existing states.

**(3) Hardware Standby Mode:** A transition to hardware standby mode is made when the $\overline{\text{STBY}}$ pin goes low. In hardware standby mode, the CPU and clock halt and all MCU operations stop. The on-chip supporting modules are reset, but as long as a specified voltage is supplied, on-chip RAM contents are retained.

## 2.9 Basic Timing

### 2.9.1 Overview

The CPU is driven by a system clock, denoted by the symbol ø. The period from one rising edge of ø to the next is referred to as a "state." The memory cycle or bus cycle consists of one, two, or three states. Different methods are used to access on-chip memory, on-chip supporting modules, and the external address space.

### 2.9.2 On-Chip Memory (ROM, RAM)

On-chip memory is accessed in one state. The data bus is 16 bits wide, permitting both byte and word transfer instruction. Figure 2.17 shows the on-chip memory access cycle. Figure 2.18 shows the pin states.

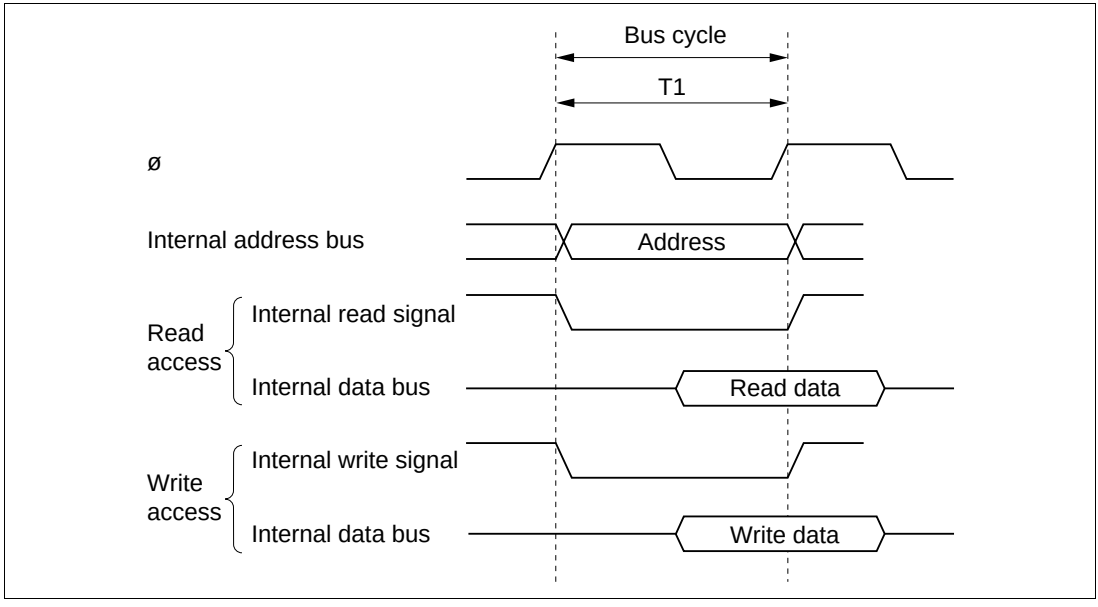



**Figure 2.17 On-Chip Memory Access Cycle**

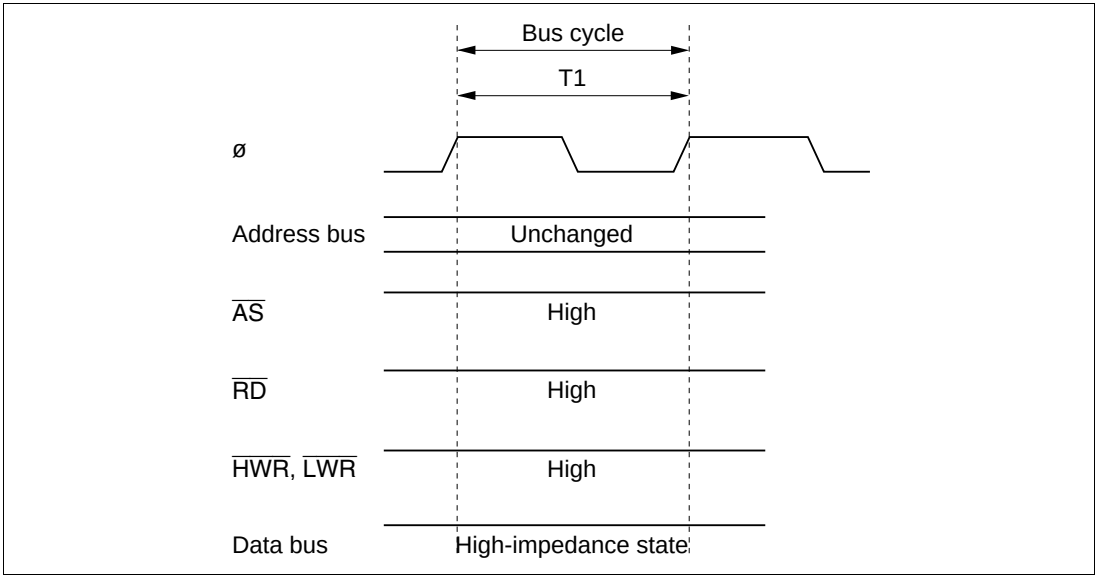



**Figure 2.18 Pin States during On-Chip Memory Access**

### 2.9.3 On-Chip Supporting Module Access Timing

The on-chip supporting modules are accessed in two states. The data bus is either 8 bits or 16 bits wide, depending on the particular internal I/O register being accessed. Figure 2.19 shows the access timing for the on-chip supporting modules. Figure 2.20 shows the pin states.

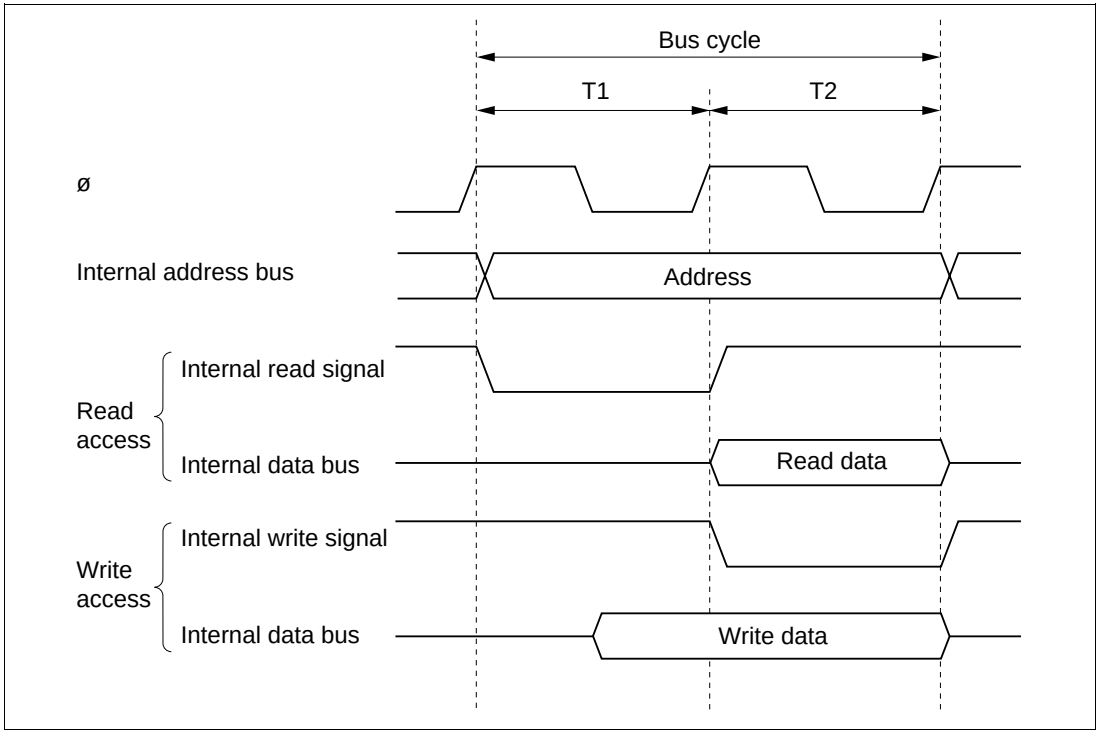



**Figure 2.19 On-Chip Supporting Module Access Cycle**

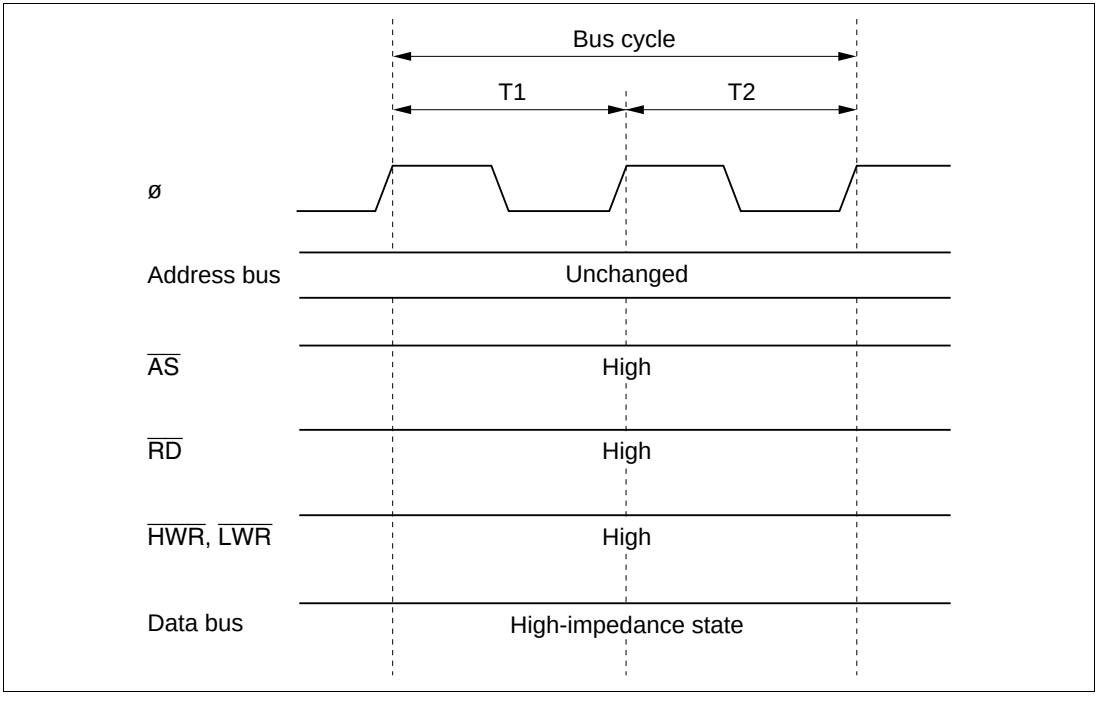



**Figure 2.20 Pin States during On-Chip Supporting Module Access**

### 2.9.4 External Address Space Access Timing

The external address space is accessed with an 8-bit or 16-bit data bus width in a two-state or three-state bus cycle. In three-state access, wait states can be inserted. For further details, refer to section 6, Bus Controller.

# Section 3 MCU Operating Modes

## 3.1 Overview

### 3.1.1 Operating Mode Selection (F-ZTAT™ Version)

The H8S/2345 Series has eight operating modes (modes 4 to 7, 10, 11, 14 and 15). These modes are determined by the mode pin ($MD_2$ to $MD_0$) and flash write enable pin (FWE) settings. The CPU operating mode and initial bus width can be selected as shown in table 3.1.

Table 3.1 lists the MCU operating modes.

**Table 3.1 MCU Operating Mode Selection (F-ZTAT™ Version)**

| MCU Operating Mode | FWE | $MD_2$ | $MD_1$ | $MD_0$ | CPU Operating Mode | Description | On-Chip ROM | External Data Bus | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | Initial Width | Max. Width |
| 0 | 0 | 0 | 0 | 0 | — | — | — | — | — |
| 1 | | | | 1 | | | | | |
| 2 | | | 1 | 0 | | | | | |
| 3 | | | | 1 | | | | | |
| 4 | | 1 | 0 | 0 | Advanced | On-chip ROM disabled, expanded mode | Disabled | 16 bits | 16 bits |
| 5 | | | | 1 | | | | 8 bits | 16 bits |
| 6 | | | 1 | 0 | | On-chip ROM enabled, expanded mode | Enabled | 8 bits | 16 bits |
| 7 | | | | 1 | | Single-chip mode | | — | — |
| 8 | 1 | 0 | 0 | 0 | — | — | — | — | — |
| 9 | | | | 1 | | | | | |
| 10 | | | 1 | 0 | Advanced | Boot mode | Enabled | 8 bits | 16 bits |
| 11 | | | | 1 | | | | — | — |
| 12 | | 1 | 0 | 0 | — | — | — | — | — |
| 13 | | | | 1 | | | | | |
| 14 | | | 1 | 0 | Advanced | User program mode | Enabled | 8 bits | 16 bits |
| 15 | | | | 1 | | | | — | — |

The CPU's architecture allows for 4 Gbytes of address space, but the H8S/2345 Series actually accesses a maximum of 16 Mbytes.

Modes 4 to 6 are externally expanded modes that allow access to external memory and peripheral devices.

The external expansion modes allow switching between 8-bit and 16-bit bus modes. After program execution starts, an 8-bit or 16-bit address space can be set for each area, depending on the bus controller setting. If 16-bit access is selected for any one area, 16-bit bus mode is set; if 8-bit access is selected for all areas, 8-bit bus mode is set.

Note that the functions of each pin depend on the operating mode.

Modes 10, 11, 14, and 15 are boot modes and user program modes in which the flash memory can be programmed and erased. For details, see section 17, ROM.

The H8S/2345 Series can only be used in modes 4 to 7, 10, 11, 14, and 15. This means that the flash write enable pin and mode pins must be set to select one of these modes.

Do not change the inputs at the mode pins during operation.

### 3.1.2 Operating Mode Selection (ZTAT, Mask ROM, and ROMless Versions)

The H8S/2345 Series has seven operating modes (modes 1 to 7). These modes enable selection of the CPU operating mode, enabling/disabling of on-chip ROM, and the initial bus width setting, by setting the mode pins ($MD_2$ to $MD_0$).

Table 3.2 lists the MCU operating modes.

**Table 3.2 MCU Operating Mode Selection**

| MCU Operating Mode | $MD_2$ | $MD_1$ | $MD_0$ | CPU Operating Mode | Description | On-Chip ROM | External Data Bus | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | Initial Width | Max. Width |
| 0 | 0 | 0 | 0 | — | — | — | — | |
| 1 | | | 1 | Normal | On-chip ROM disabled, expanded mode | Disabled | 8 bits | 16 bits |
| 2* | | 1 | 0 | | On-chip ROM enabled, expanded mode | Enabled | 8 bits | 16 bits |
| 3* | | | 1 | | Single-chip mode | | — | |
| 4 | 1 | 0 | 0 | Advanced | On-chip ROM disabled, expanded mode | Disabled | 16 bits | 16 bits |
| 5 | | | 1 | | | | 8 bits | 16 bits |
| 6* | | 1 | 0 | | On-chip ROM enabled, expanded mode | Enabled | 8 bits | 16 bits |
| 7* | | | 1 | | Single-chip mode | | — | |

Note: * Not used on ROMless version.

The CPU's architecture allows for 4 Gbytes of address space, but the H8S/2345 Series actually accesses a maximum of 16 Mbytes.

Modes 1, 2, and 4 to 6 are externally expanded modes that allow access to external memory and peripheral devices.

The external expansion modes allow switching between 8-bit and 16-bit bus modes. After program execution starts, an 8-bit or 16-bit address space can be set for each area, depending on the bus controller setting. If 16-bit access is selected for any one area, 16-bit bus mode is set; if 8-bit access is selected for all areas, 8-bit bus mode is set.

Note that the functions of each pin depend on the operating mode.

The H8S/2345 Series can be used only in modes 1 to 7. This means that the mode pins must be set to select one of these modes. Do not change the inputs at the mode pins during operation.

### 3.1.3 Register Configuration

The H8S/2345 Series has a mode control register (MDCR) that indicates the inputs at the mode pins ($MD_2$ to $MD_0$), and a system control register (SYSCR) and a system control register 2 (SYSCR2)*[2] that control the operation of the H8S/2345 Series. Table 3.3 summarizes these registers.

**Table 3.3 MCU Registers**

| Name | Abbreviation | R/W | Initial Value | Address*[1] |
|---|---|---|---|---|
| Mode control register | MDCR | R | Undetermined | H'FF3B |
| System control register | SYSCR | R/W | H'01 | H'FF39 |
| System control register 2*[2] | SYSCR2 | R/W | H'00 | H'FF42 |

Notes: 1. Lower 16 bits of the address.
2. The SYSCR2 register can only be used in the F-ZTAT version. In the ZTAT, mask ROM, and ROMless versions, this register cannot be written to and will return an undefined value of read.

## 3.2 Register Descriptions

### 3.2.1 Mode Control Register (MDCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | — | MDS2 | MDS1 | MDS0 |
| Initial value: | | 1 | 0 | 0 | 0 | 0 | —* | —* | —* |
| R/W | : | — | — | — | — | — | R | R | R |

Note: * Determined by pins $MD_2$ to $MD_0$.

MDCR is an 8-bit read-only register that indicates the current operating mode of the H8S/2345 Series.

**Bit 7—Reserved:** Read-only bit, always read as 1.

**Bits 6 to 3—Reserved:** Read-only bits, always read as 0.

**Bits 2 to 0—Mode Select 2 to 0 (MDS2 to MDS0):** These bits indicate the input levels at pins $MD_2$ to $MD_0$ (the current operating mode). Bits MDS2 to MDS0 correspond to $MD_2$ to $MD_0$. MDS2 to MDS0 are read-only bits-they cannot be written to. The mode pin ($MD_2$ to $MD_0$) input levels are latched into these bits when MDCR is read. These latches are canceled by a power-on reset, but are retained after a manual reset.

### 3.2.2 System Control Register (SYSCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | INTM1 | INTM0 | NMIEG | — | — | RAME |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Bits 7 and 6—Reserved:** Only 0 should be written to these bits.

**Bits 5 and 4—Interrupt Control Mode 1 and 0 (INTM1, INTM0):** These bits select the control mode of the interrupt controller. For details of the interrupt control modes, see section 5.4.1, Interrupt Control Modes and Interrupt Operation.

| Bit 5 INTM1 | Bit 4 INTM0 | Interrupt Control Mode | Description | |
|---|---|---|---|---|
| 0 | 0 | 0 | Control of interrupts by I bit | (Initial value) |
| | 1 | — | Setting prohibited | |
| 1 | 0 | 2 | Control of interrupts by I2 to I0 bits and IPR | |
| | 1 | — | Setting prohibited | |

**Bit 3—NMI Edge Select (NMIEG):** Selects the valid edge of the NMI interrupt input.

| Bit 3 NMIEG | Description | |
|---|---|---|
| 0 | An interrupt is requested at the falling edge of NMI input | (Initial value) |
| 1 | An interrupt is requested at the rising edge of NMI input | |

**Bits 2 and 1—Reserved:** Only 0 should be written to these bits.

**Bit 0—RAM Enable (RAME):** Enables or disables the on-chip RAM. The RAME bit is initialized when the reset status is released. It is not initialized in software standby mode.

| Bit 0 RAME | Description | |
|---|---|---|
| 0 | On-chip RAM is disabled | |
| 1 | On-chip RAM is enabled | (Initial value) |

### 3.2.3 System Control Register 2 (SYSCR2) (F-ZTAT Version Only)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | FLSHE | — | — | — |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | — | — | R/W | — | — | — |

SYSCR2 is an 8-bit readable/writable register that performs on-chip flash memory control.

SYSCR2 is initialized to H'00 by a reset and in hardware standby mode.

SYSCR2 can only be accessed in the F-ZTAT version. In other versions, this register cannot be written to and will return an undefined value if read.

**Bits 7 to 4—Reserved:** Read-only bits, always read as 0.

**Bit 3—Flash Memory Control Register Enable (FLSHE):** Controls CPU access to the flash memory control registers (FLMCR1, FLMCR2, EBR1, and EBR2). For details, see section 17, ROM.

**Bit 3**

| FLSHE | Description |
|---|---|
| 0 | Flash control registers are not selected for addresses H'FFFFC8 to H'FFFFCB (Initial value) |
| 1 | Flash control registers are selected for addresses H'FFFFC8 to H'FFFFCB |

**Bits 2 to 0—Reserved:** Read-only bits, always read as 0.

## 3.3 Operating Mode Descriptions

### 3.3.1 Mode 1 (ZTAT, Mask ROM, and ROMless Versions Only)

The CPU can access a 64-kbyte address space in normal mode. The on-chip ROM is disabled, and 8-bit bus mode is set, immediately after a reset.

Ports B and C function as an address bus, port D functions as a data bus, and part of port F carries bus control signals. However, note that if 16-bit access is designated by the bus controller, the bus mode switches to 16 bits and port E becomes a data bus.

### 3.3.2 Mode 2*[1] (ZTAT and Mask ROM Versions Only)

The CPU can access a 64-kbyte address space in normal mode. The on-chip ROM is enabled, and 8-bit bus mode is set. immediately after a reset.

Ports B and C function as input ports immediately after a reset. They can each be set to output addresses by setting the corresponding bits in the data direction register (DDR) to 1. Port D functions as a data bus, and part of port F carries bus control signals. However, note that if 16-bit access is designated by the bus controller, the bus mode switches to 16 bits and port E becomes a data bus.

The amount of on-chip ROM that can be used is limited to 56 kbytes.

### 3.3.3 Mode 3*[1] (ZTAT and Mask ROM Versions Only)

The CPU can access a 64-kbyte address space in normal mode. The on-chip ROM is enabled, but external addresses cannot be accessed.

All I/O ports are available for use as input-output ports.

The amount of on-chip ROM that can be used is limited to 56 kbytes.

### 3.3.4 Mode 4*[2]

The CPU can access a 16-Mbyte address space in advanced mode. The on-chip ROM is disabled.

Pins $P1_3$ to $P1_0$, ports A, B and C function as an address bus, ports D and E function as a data bus, and part of port F carries bus control signals. Pins $P1_3$ to $P1_0$ function as inputs immediately after a reset. Each of these pins can be set to output addresses by setting the corresponding bit in the data direction register (DDR) to 1.

The initial bus mode after a reset is 16 bits, with 16-bit access to all areas. However, note that if 8-bit access is designated by the bus controller for all areas, the bus mode switches to 8 bits.

### 3.3.5 Mode 5*[2]

The CPU can access a 16-Mbyte address space in advanced mode. The on-chip ROM is disabled.

Pins $P1_3$ to $P1_0$, ports A, B and C function as an address bus, port D function as a data bus, and part of port F carries bus control signals. Pins $P1_3$ to $P1_0$ function as inputs immediately after a reset. They can each be set to output addresses by setting the corresponding bits in the data direction register (DDR) to 1.

The initial bus mode after a reset is 8 bits, with 8-bit access to all areas. However, note that if at least one area is designated for 16-bit access by the bus controller, the bus mode switches to 16 bits and port E becomes a data bus.

### 3.3.6 Mode 6*[1]

The CPU can access a 16-Mbyte address space in advanced mode. The on-chip ROM is enabled.

Pins $P1_3$ to $P1_0$, ports A, B and C function as input ports immediately after a reset. They can each be set to output addresses by setting the corresponding bits in the data direction register (DDR) to 1. Port D functions as a data bus, and part of port F carries bus control signals.

The initial bus mode after a reset is 8 bits, with 8-bit access to all areas. However, if any area is designated as 16-bit access space by the bus controller, 16-bit bus mode is set and port E becomes a data bus.

### 3.3.7 Mode 7*[1]

The CPU can access a 16-Mbyte address space in advanced mode. The on-chip ROM is enabled, but external addresses cannot be accessed.

All I/O ports are available for use as input-output ports.

Notes: 1. Not used on ROMless version.
2. The upper address pins ($A_{23}$ to $A_{20}$) cannot be used as outputs in modes 4 or 5 immediately after a reset. To use the upper address pins ($A_{23}$ to $A_{20}$) as outputs, it is necessary to first set the corresponding bits in the port 1 data direction register (P1DDR) to 1.

### 3.3.8 Modes 8 and 9 (F-ZTAT Version Only)

Modes 8 and 9 are not supported in the H8S/2345 Series, and must not be set.

### 3.3.9 Mode 10 (F-ZTAT Version Only)

This is a flash memory boot mode. For details, see section 17, ROM.

MCU operation is the same as in mode 6.

### 3.3.10 Mode 11 (F-ZTAT Version Only)

This is a flash memory boot mode. For details, see section 17, ROM.

MCU operation is the same as in mode 7.

### 3.3.11 Modes 12 and 13 (F-ZTAT Version Only)

Modes 12 and 13 are not supported in the H8S/2345 Series, and must not be set.

### 3.3.12 Mode 14 (F-ZTAT Version Only)

This is a flash memory user program mode. For details, see section 17, ROM.

MCU operation is the same as in mode 6.

### 3.3.13 Mode 15 (F-ZTAT Version Only)

This is a flash memory user program mode. For details, see section 17, ROM.

MCU operation is the same as in mode 7.

## 3.4 Pin Functions in Each Operating Mode

The pin functions of ports 1 and A to F vary depending on the operating mode. Table 3.3 shows their functions in each operating mode.

**Table 3.3 Pin Functions in Each Mode**

| Port | | Mode 1*[2] | Mode 2*[3] | Mode 3*[3] | Mode 4 | Mode 5 | Mode 6*[3] Mode 10*[4] Mode 14*[4] | Mode 7*[3] Mode 11*[4] Mode 15*[4] |
|---|---|---|---|---|---|---|---|---|
| Port 1 | $P1_3$ to $P1_0$ | P*[1]/T | P*[1]/T | P*[1]/T | P*[1]/T/A | P*[1]/T/A | P*[1]/T/A | P*[1]/T |
| Port A | $PA_3$ to $PA_0$ | P | P | P | A | A | P*[1]/A | P |
| Port B | | A | P*[1]/A | P | A | A | P*[1]/A | P |
| Port C | | A | P*[1]/A | P | A | A | P*[1]/A | P |
| Port D | | D | D | P | D | D | D | P |
| Port E | | P*[1]/D | P*[1]/D | P | P/D | P*[1]/D | P*[1]/D | P |
| Port F | $PF_7$ | P/C*[1] | P/C*[1] | P*[1]/C | P/C*[1] | P/C*[1] | P/C*[1] | P*[1]/C |
| | $PF_6$ to $PF_3$ | C | C | P | C | C | C | P |
| | $PF_2$ to $PF_0$ | P*[1]/C | P*[1]/C | | P*[1]/C | P*[1]/C | P*[1]/C | |

**Legend**

P: I/O port

T: Timer I/O

A: Address bus output

D: Data bus I/O

C: Control signals, clock I/O

Notes: 1. After reset
2. Not used on F-ZTAT.
3. Not used on ROMless version.
4. Applies to F-ZTAT version only.

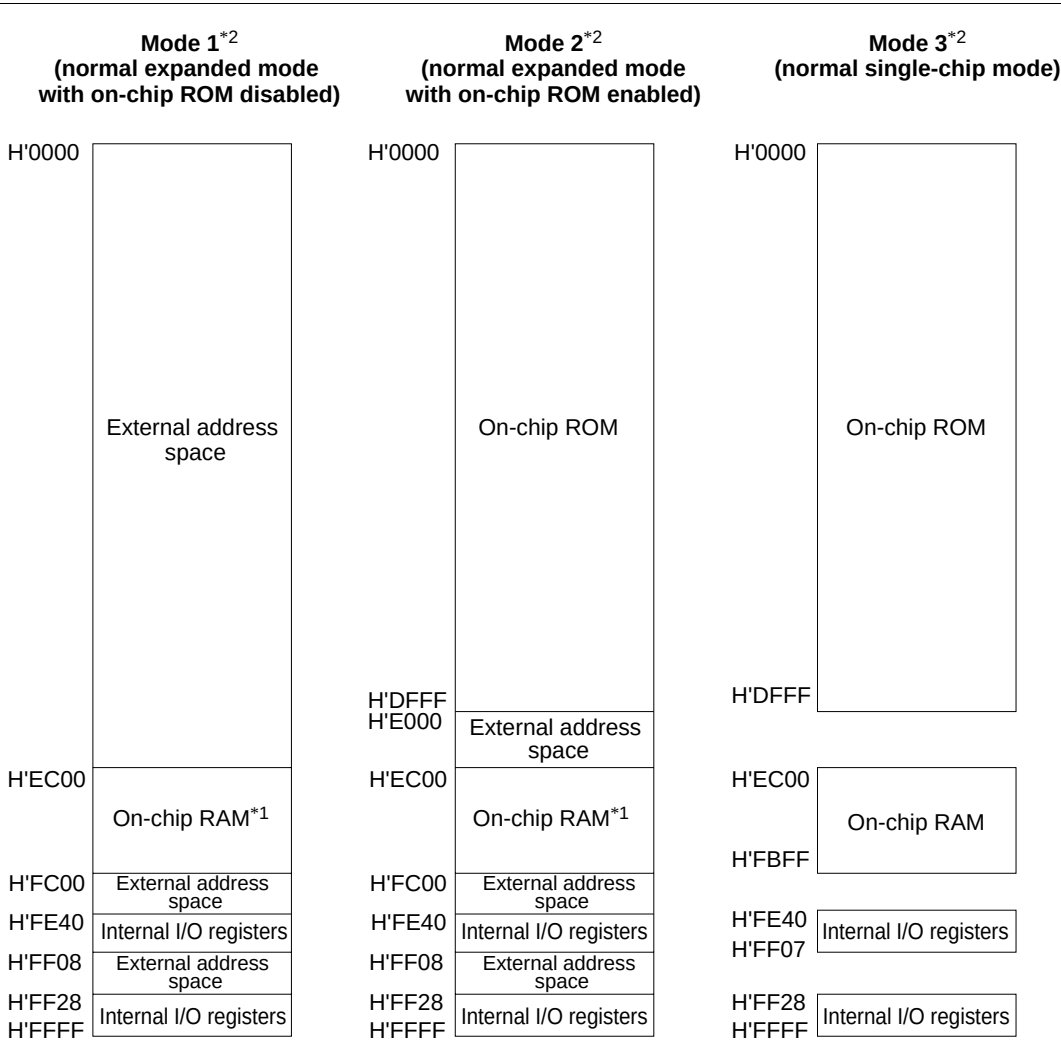
## 3.5 Memory Map in Each Operating Mode

Memory maps for the H8S/2345, H8S/2344, H8S/2343, H8S/2341, and H8S/2340 are shown in figure 3.1 to figure 3.5.

The address space is 64 kbytes in modes 1 to 3 (normal modes)*, and 16 Mbytes in modes 4 to 7, 10, 11, 14, and 15 (advanced modes). The on-chip ROM capacity of the H8S/2345 is 128 kbytes, that of the H8S/2344 96 kbytes, and that of the H8S/2343 64 kbytes. However, only 56 kbytes are available in modes 2 and 3 (normal modes)*.

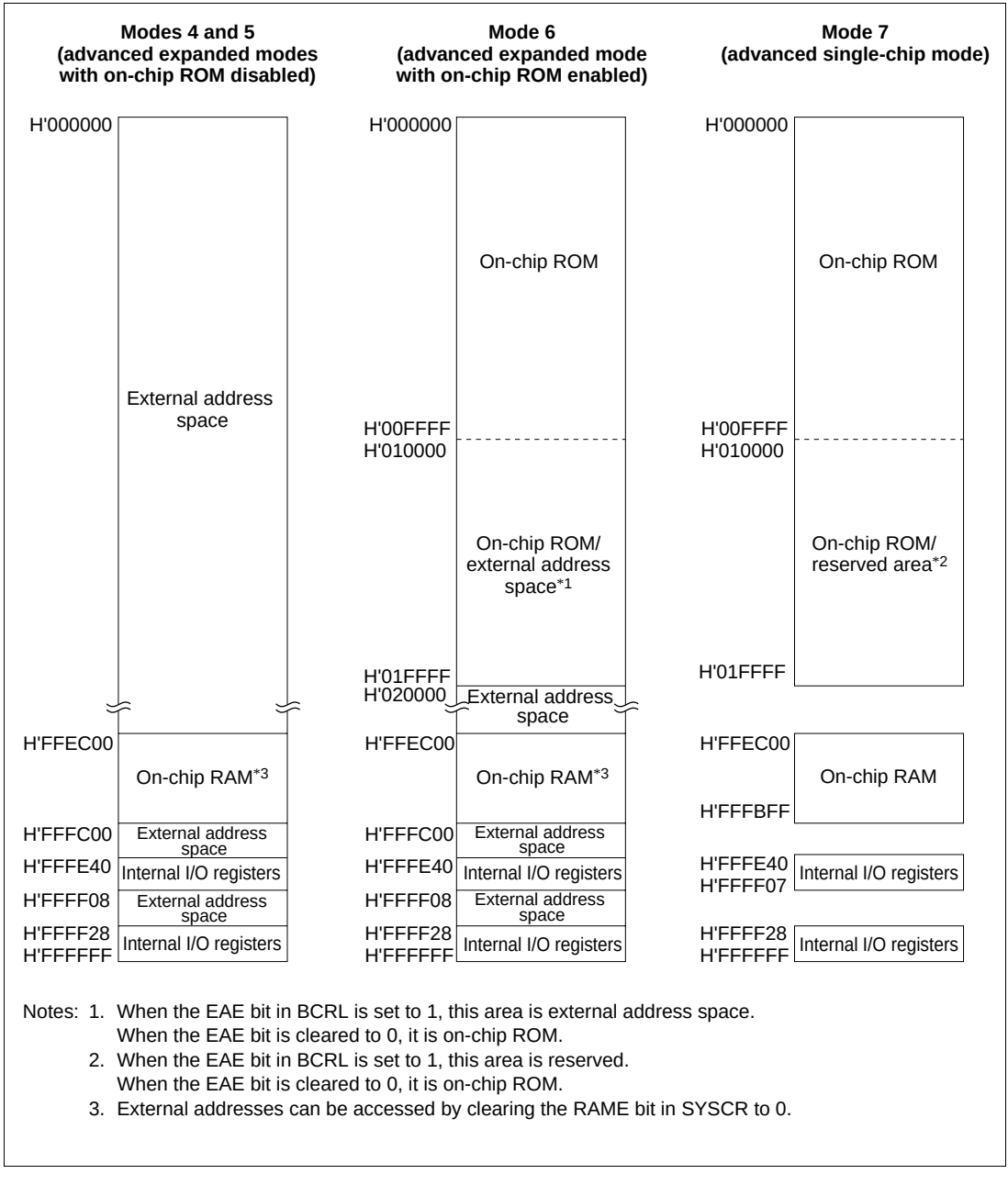
The address space is divided into eight areas for modes 4 to 6, 10, and 14. For details, see section 6, Bus Controller.

Note: * Not available on F-ZTAT version.



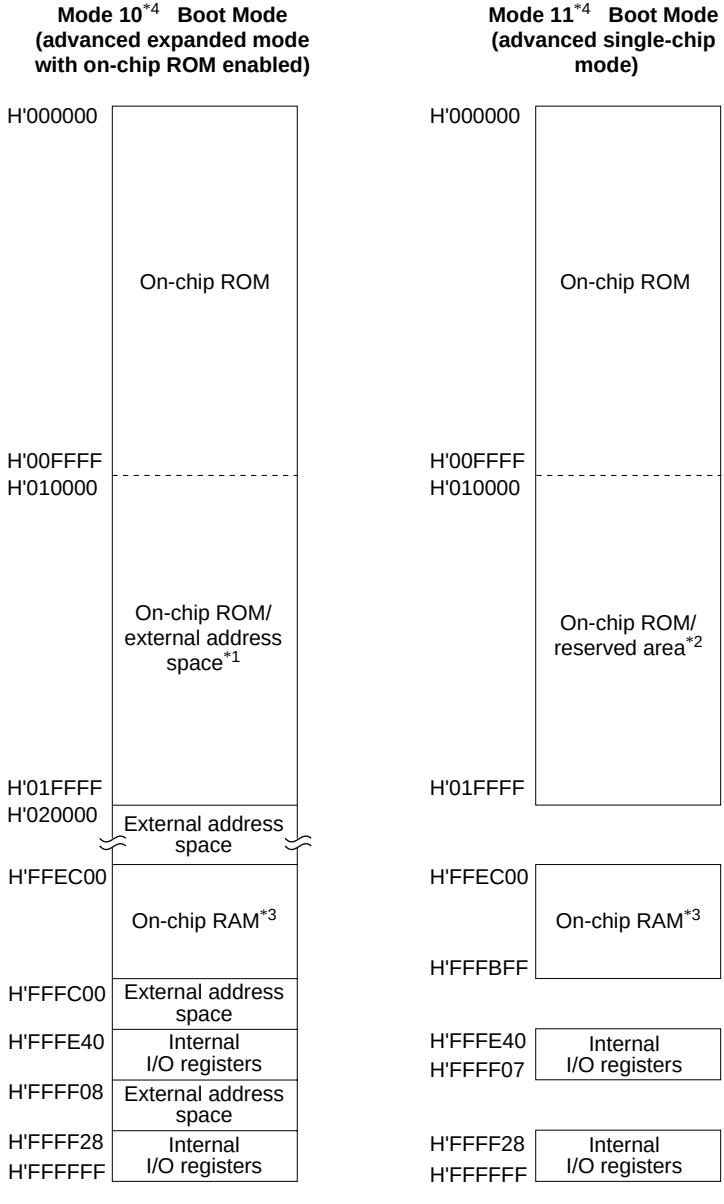

Notes: 1. External addresses can be accessed by clearing the RAME bit in SYSCR to 0.
2. Not available on F-ZTAT version.

**Figure 3.1 Memory Map in Each Operating Mode in the H8S/2345**



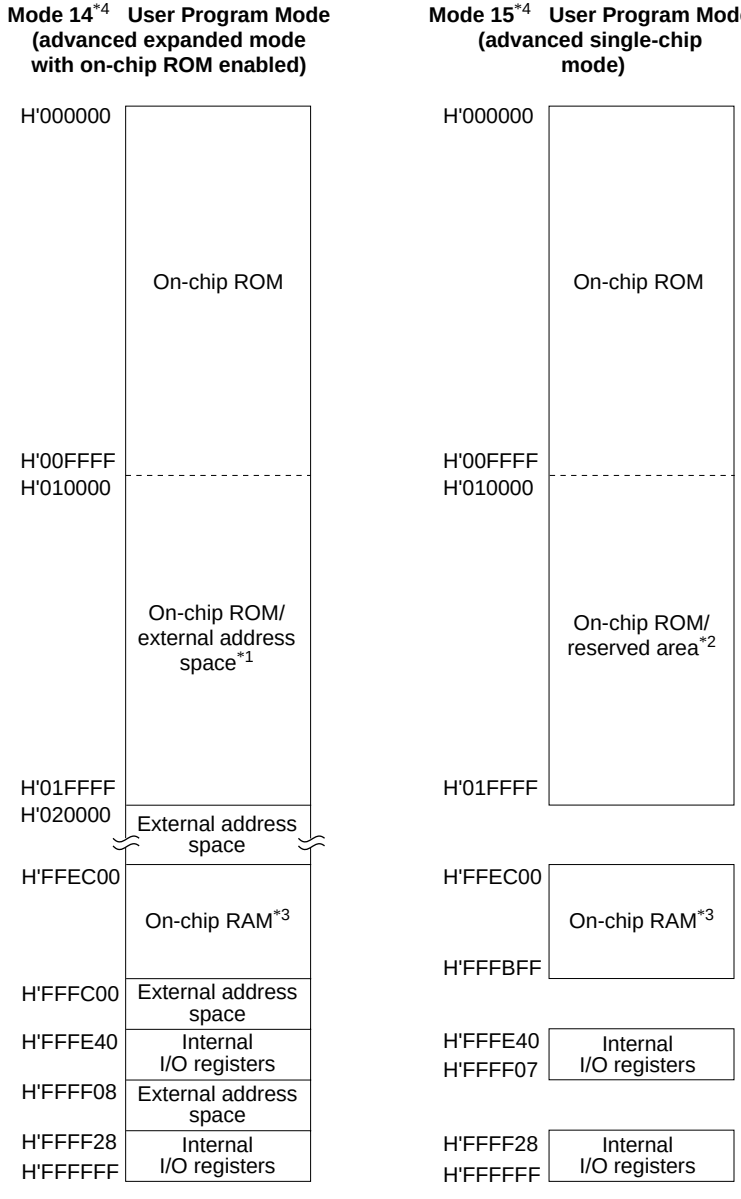

Notes: 1. When the EAE bit in BCRL is set to 1, this area is external address space.
When the EAE bit is cleared to 0, it is on-chip ROM.
2. When the EAE bit in BCRL is set to 1, this area is reserved.
When the EAE bit is cleared to 0, it is on-chip ROM.
3. External addresses can be accessed by clearing the RAME bit in SYSCR to 0.

**Figure 3.1 Memory Map in Each Operating Mode in the H8S/2345 (cont)**




Notes: 1. When the EAE bit in BCRL is set to 1, this area is external address space.
When the EAE bit is cleared to 0, it is on-chip ROM.
2. When the EAE bit in BCRL is set to 1, this area is reserved.
When the EAE bit is cleared to 0, it is on-chip ROM.
3. On-chip RAM is used for flash memory programming. Do not clear the RAME bit to 0 in SYSCR.
4. Modes 10 and 11 are provided in the F-ZTAT version only.

**Figure 3.1 Memory Map in Each Operating Mode in the H8S/2345 (cont)**




Notes: 1. When the EAE bit in BCRL is set to 1, this area is external address space. When the EAE bit is cleared to 0, it is on-chip ROM.
2. When the EAE bit in BCRL is set to 1, this area is reserved. When the EAE bit is cleared to 0, it is on-chip ROM.
3. On-chip RAM is used for flash memory programming. Do not clear the RAME bit to 0 in SYSCR.
4. Modes 14 and 15 are provided in the F-ZTAT version only.

**Figure 3.1 Memory Map in Each Operating Mode in the H8S/2345 (cont)**

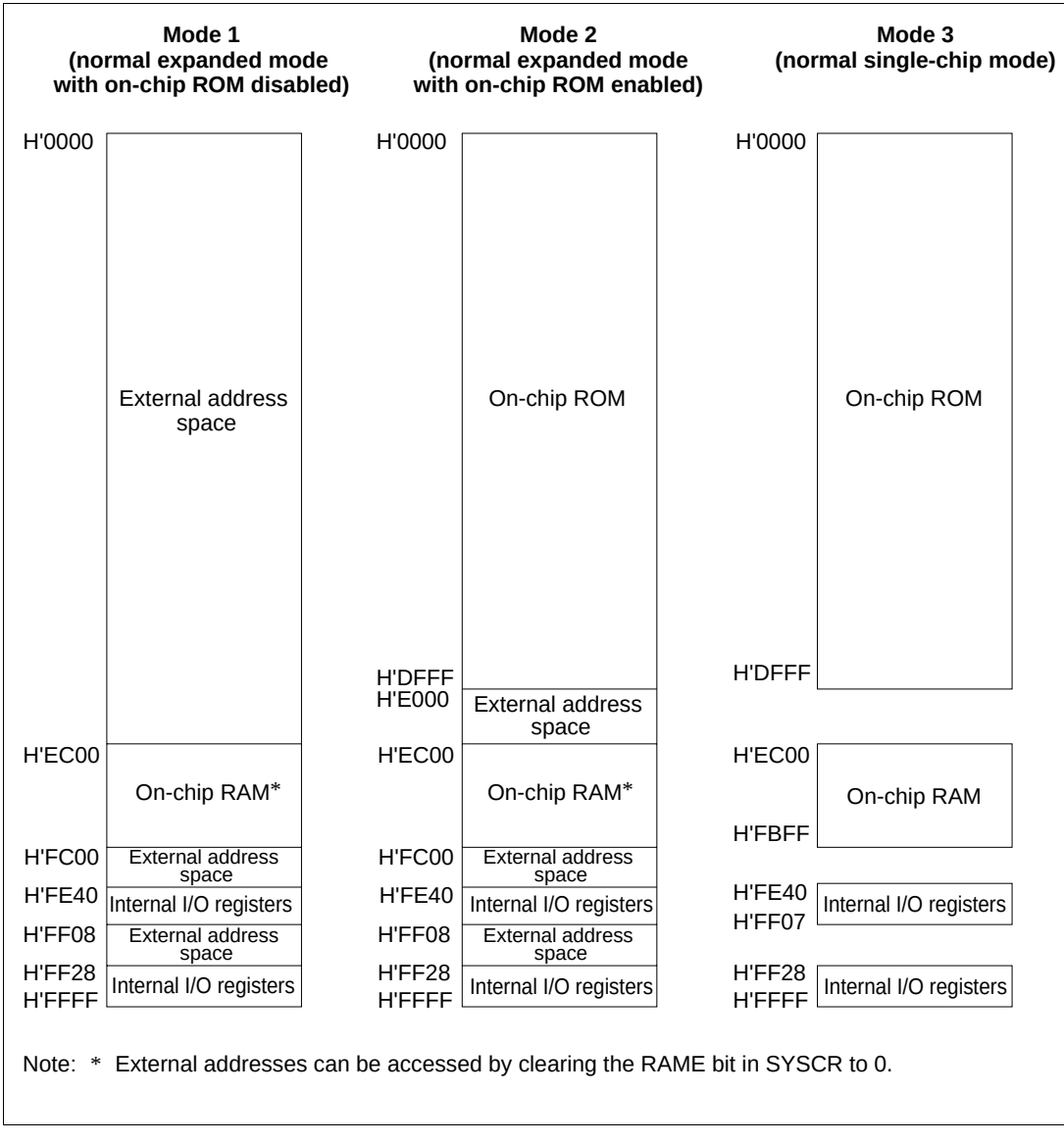
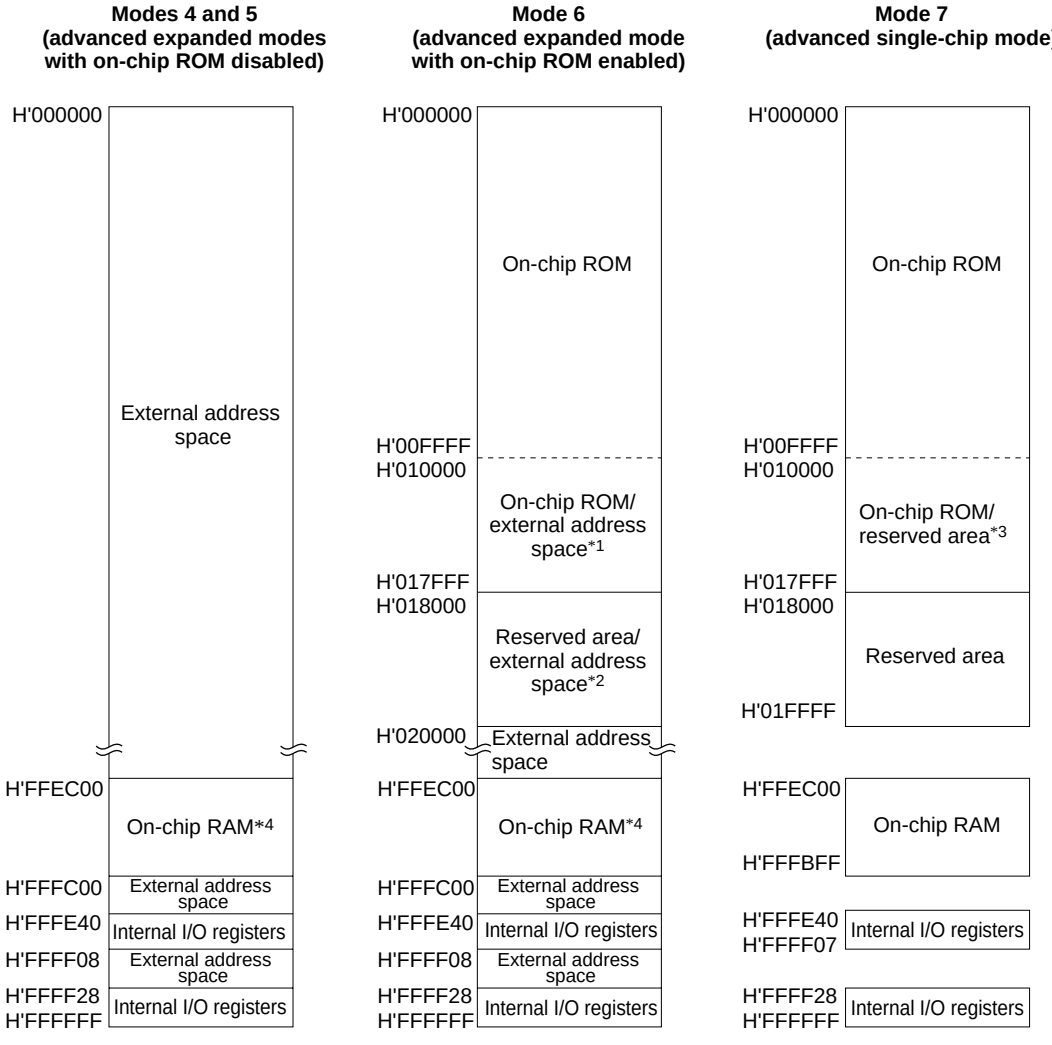
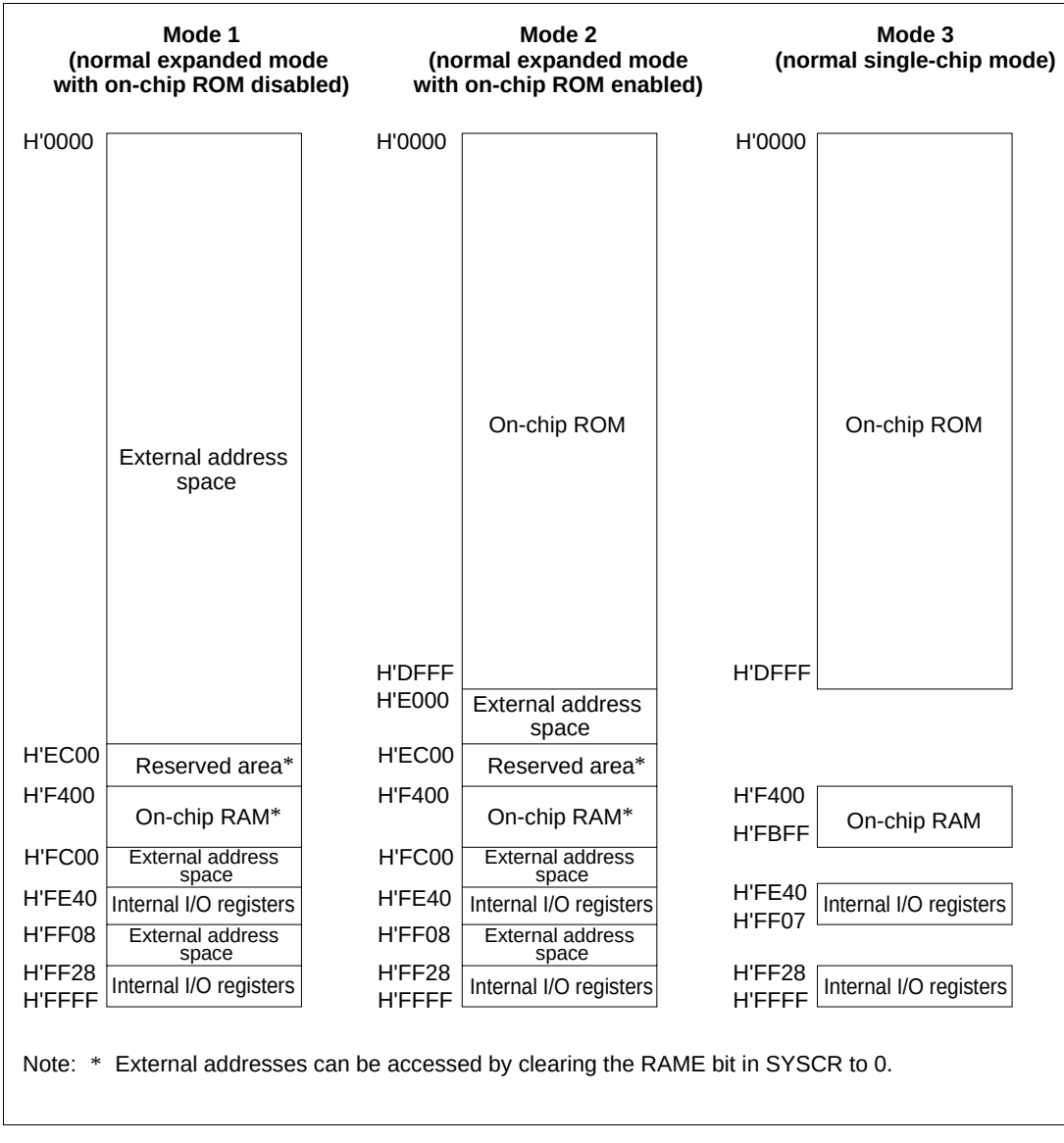



**Figure 3.2 Memory Map in Each Operating Mode in the H8S/2344**



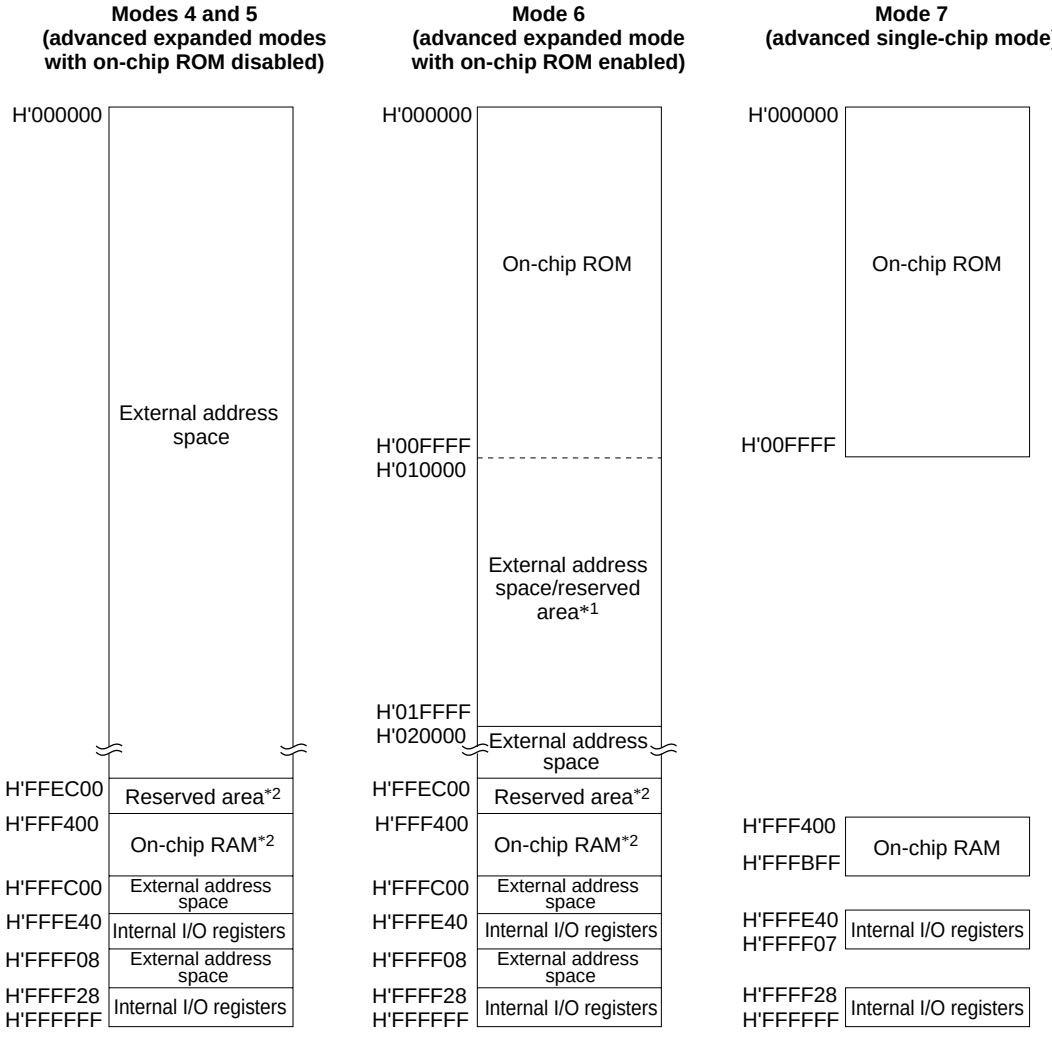

Notes: 1. When the EAE bit in BCRL is set to 1, this area is external address space.
When the EAE bit is cleared to 0, it is on-chip ROM.
2. When the EAE bit in BCRL is set to 1, this area is external address space.
When the EAE bit is cleared to 0, it is a reserved area.
3. This area is reserved when the EAE bit in BCRL is set to 1, and on-chip ROM when the EAE bit is cleared to 0.
4. External addresses can be accessed by clearing the RAME bit in SYSCR to 0.

**Figure 3.2 Memory Map in Each Operating Mode in the H8S/2344 (cont)**



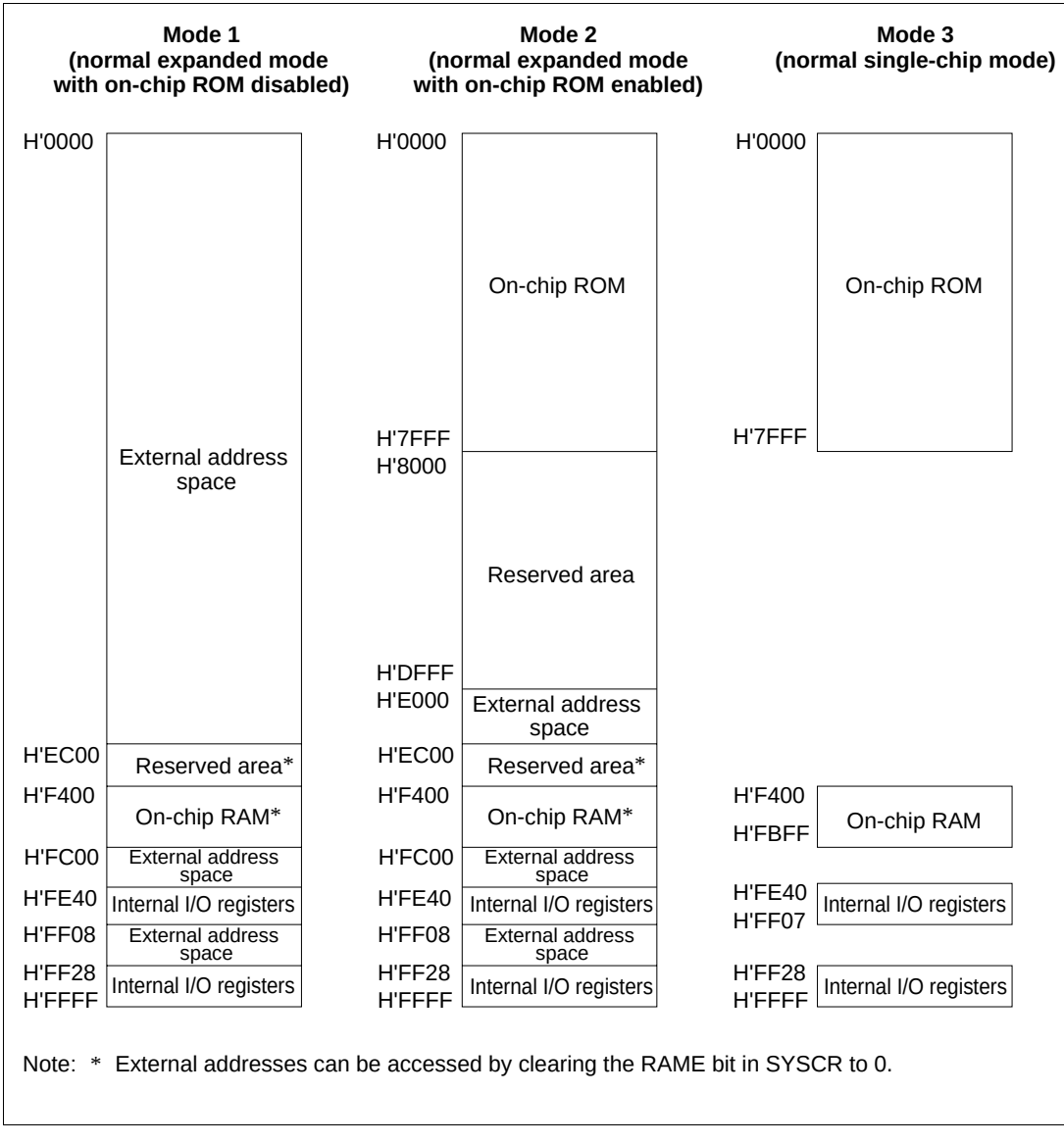

Note: * External addresses can be accessed by clearing the RAME bit in SYSCR to 0.

**Figure 3.3 Memory Map in Each Operating Mode in the H8S/2343**



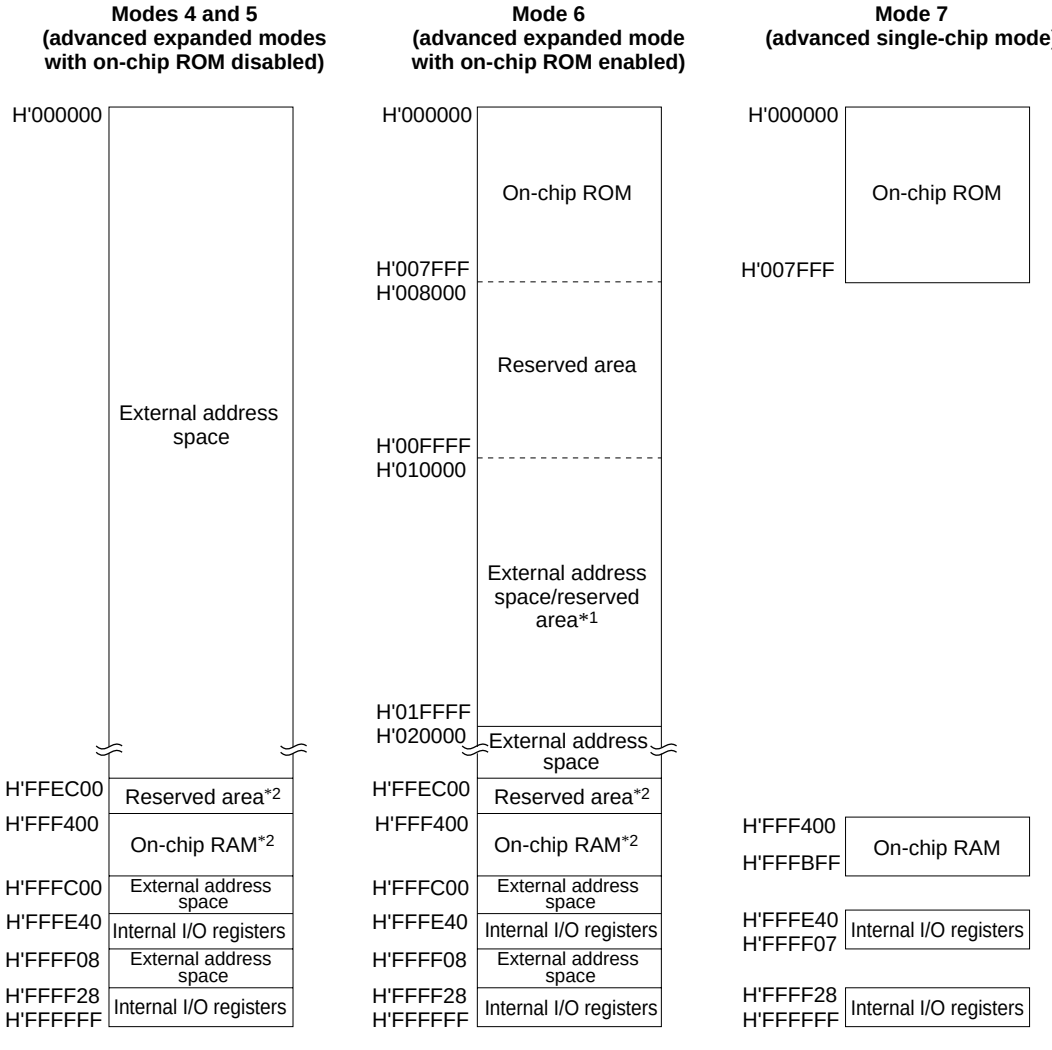

Notes: 1. When the EAE bit in BCRL is set to 1, this area is external address space. When the EAE bit is cleared to 0, it is on-chip ROM.
2. External addresses can be accessed by clearing the RAME bit in SYSCR to 0.

**Figure 3.3 Memory Map in Each Operating Mode in the H8S/2343 (cont)**




Note: * External addresses can be accessed by clearing the RAME bit in SYSCR to 0.

**Figure 3.4 Memory Map in Each Operating Mode in the H8S/2341**




Notes: 1. When the EAE bit in BCRL is set to 1, this area is external address space. When the EAE bit is cleared to 0, it is on-chip ROM.
2. External addresses can be accessed by clearing the RAME bit in SYSCR to 0.

**Figure 3.4 Memory Map in Each Operating Mode in the H8S/2341 (cont)**

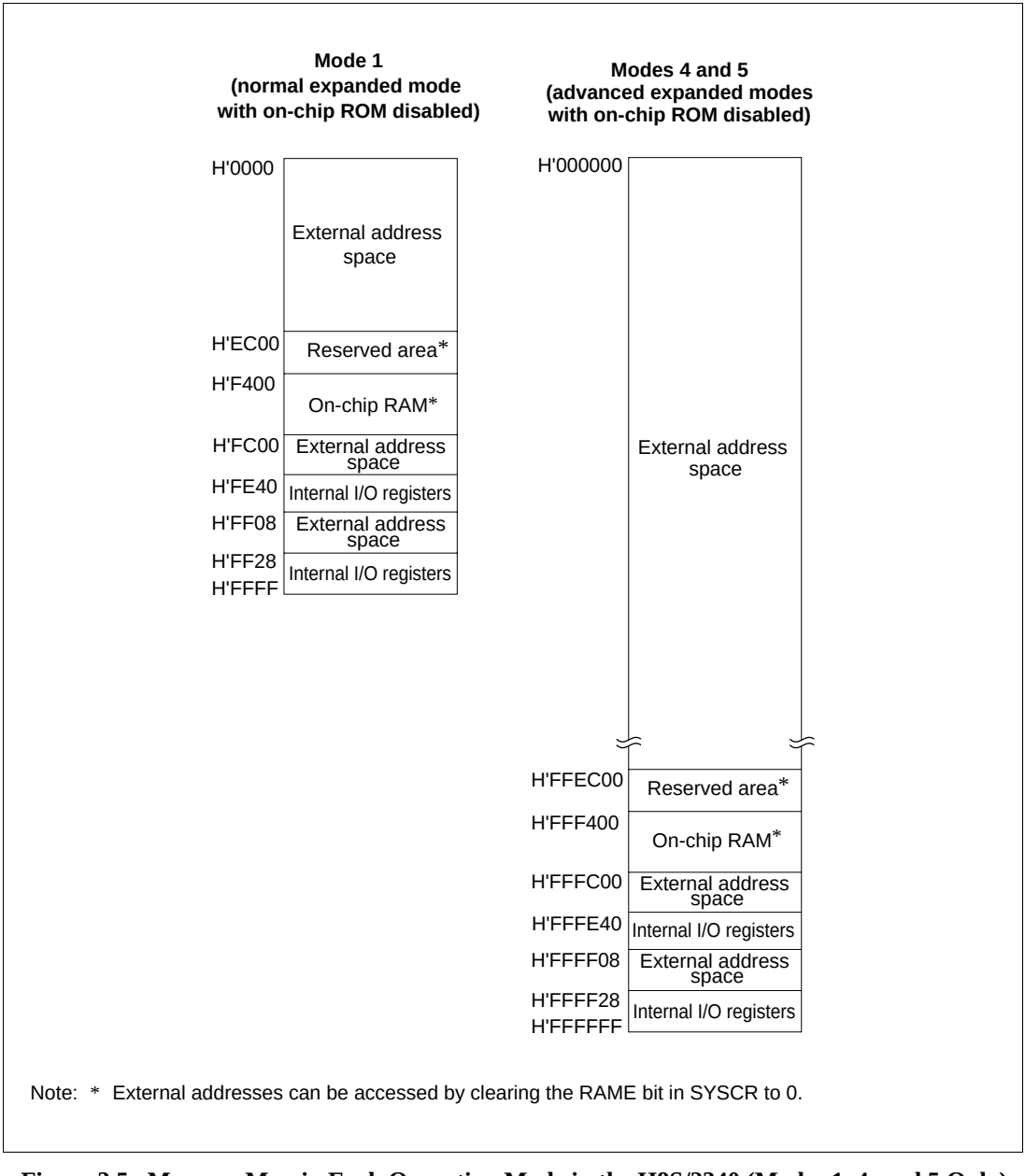



**Figure 3.5 Memory Map in Each Operating Mode in the H8S/2340 (Modes 1, 4, and 5 Only)**

# Section 4 Exception Handling

## 4.1 Overview

### 4.1.1 Exception Handling Types and Priority

As table 4.1 indicates, exception handling may be caused by a reset, trap instruction, or interrupt. Exception handling is prioritized as shown in table 4.1. If two or more exceptions occur simultaneously, they are accepted and processed in order of priority. Trap instruction exceptions are accepted at all times, in the program execution state.

Exception handling sources, the stack structure, and the operation of the CPU vary depending on the interrupt control mode set by the INTM0 and INTM1 bits of SYSCR.

**Table 4.1 Exception Types and Priority**

| Priority | Exception Type | Start of Exception Handling |
|---|---|---|
| High | Reset | Starts immediately after a low-to-high transition at the $\overline{\text{RES}}$ pin, or when the watchdog timer overflows. The CPU enters the power-on reset state when the NMI pin is high, or the manual reset state when the NMI pin is low. |
| | Trace*[1] | Starts when execution of the current instruction or exception handling ends, if the trace (T) bit is set to 1 |
| | Interrupt | Starts when execution of the current instruction or exception handling ends, if an interrupt request has been issued*[2] |
| Low | Trap instruction (TRAPA)*[3] | Started by execution of a trap instruction (TRAPA) |

Notes: 1. Traces are enabled only in interrupt control mode 2. Trace exception handling is not executed after execution of an RTE instruction.

2. Interrupt detection is not performed on completion of ANDC, ORC, XORC, or LDC instruction execution, or on completion of reset exception handling.
3. Trap instruction exception handling requests are accepted at all times in program execution state.

### 4.1.2 Exception Handling Operation

Exceptions originate from various sources. Trap instructions and interrupts are handled as follows:

1. The program counter (PC), condition code register (CCR), and extended register (EXR) are pushed onto the stack.
2. The interrupt mask bits are updated. The T bit is cleared to 0.
3. A vector address corresponding to the exception source is generated, and program execution starts from that address.

For a reset exception, steps 2 and 3 above are carried out.

### 4.1.3 Exception Vector Table

The exception sources are classified as shown in figure 4.1. Different vector addresses are assigned to different exception sources.

Table 4.2 lists the exception sources and their vector addresses.

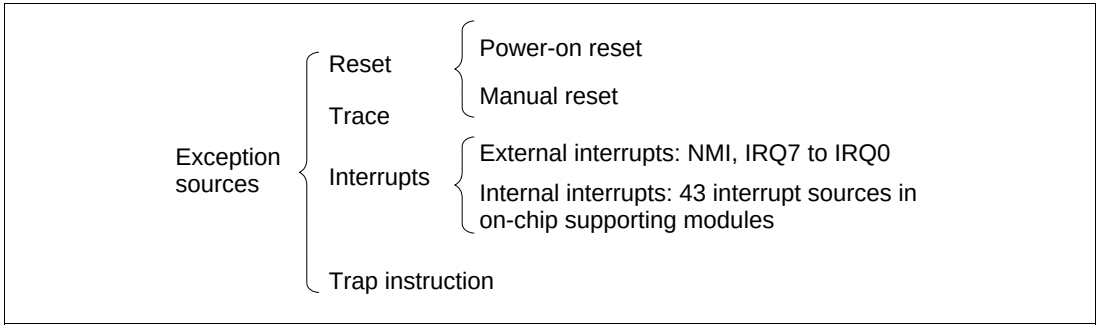



**Figure 4.1 Exception Sources**

In modes 6 and 7 in the H8S/2345, the on-chip ROM available for use after a power-on reset is the 64-kbyte area comprising addresses H'000000 to H'00FFFF. Care is required when setting vector addresses. In this case, clearing the EAE bit in BCRL enables the 128-kbyte area comprising addresses H'000000 to H'01FFFF to be used.

**Table 4.2 Exception Vector Table**

| Exception Source | | Vector Number | Vector Address*1 | |
|---|---|---|---|---|
| | | | Normal Mode*3 | Advanced Mode |
| Power-on reset | | 0 | H'0000 to H'0001 | H'0000 to H'0003 |
| Manual reset | | 1 | H'0002 to H'0003 | H'0004 to H'0007 |
| Reserved for system use | | 2 | H'0004 to H'0006 | H'0008 to H'000B |
| | | 3 | H'0006 to H'0007 | H'000C to H'000F |
| | | 4 | H'0008 to H'0009 | H'0010 to H'0013 |
| Trace | | 5 | H'000A to H'000B | H'0014 to H'0017 |
| Reserved for system use | | 6 | H'000C to H'000D | H'0018 to H'001B |
| External interrupt | NMI | 7 | H'000E to H'000F | H'001C to H'001F |
| Trap instruction (4 sources) | | 8 | H'0010 to H'0011 | H'0020 to H'0023 |
| | | 9 | H'0012 to H'0013 | H'0024 to H'0027 |
| | | 10 | H'0014 to H'0015 | H'0028 to H'002B |
| | | 11 | H'0016 to H'0017 | H'002C to H'002F |
| Reserved for system use | | 12 | H'0018 to H'0019 | H'0030 to H'0033 |
| | | 13 | H'001A to H'001B | H'0034 to H'0037 |
| | | 14 | H'001C to H'001D | H'0038 to H'003B |
| | | 15 | H'001E to H'001F | H'003C to H'003F |
| External interrupt | IRQ0 | 16 | H'0020 to H'0021 | H'0040 to H'0043 |
| | IRQ1 | 17 | H'0022 to H'0023 | H'0044 to H'0047 |
| | IRQ2 | 18 | H'0024 to H'0025 | H'0048 to H'004B |
| | IRQ3 | 19 | H'0026 to H'0027 | H'004C to H'004F |
| | IRQ4 | 20 | H'0028 to H'0029 | H'0050 to H'0053 |
| | IRQ5 | 21 | H'002A to H'002B | H'0054 to H'0057 |
| | IRQ6 | 22 | H'002C to H'002D | H'0058 to H'005B |
| | IRQ7 | 23 | H'002E to H'002F | H'005C to H'005F |
| Internal interrupt*2 | | 24<br>⋮<br>87 | H'0030 to H'0031<br>⋮<br>H'00AE to H'00AF | H'0060 to H'0063<br>⋮<br>H'015C to H'015F |

Notes: 1. Lower 16 bits of the address.
2. For details of internal interrupt vectors, see section 5.3.3, Interrupt Exception Handling Vector Table.
3. ZTAT, mask ROM, and ROMless versions only.

## 4.2 Reset

### 4.2.1 Overview

A reset has the highest exception priority.

When the $\overline{\text{RES}}$ pin goes low, all processing halts and the H8S/2345 Series enters the reset state. A reset initializes the internal state of the CPU and the registers of on-chip supporting modules. Immediately after a reset, interrupt control mode 0 is set.

Reset exception handling begins when the $\overline{\text{RES}}$ pin changes from low to high.

The level of the NMI pin at reset determines whether the type of reset is a power-on reset or a manual reset.

The H8S/2345 Series can also be reset by overflow of the watchdog timer. For details see section 11, Watchdog Timer.

### 4.2.2 Reset Types

A reset can be of either of two types: a power-on reset or a manual reset. Reset types are shown in table 4.3. A power-on reset should be used when powering on.

The internal state of the CPU is initialized by either type of reset. A power-on reset also initializes all the registers in the on-chip supporting modules, while a manual reset initializes all the registers in the on-chip supporting modules except for the bus controller and I/O ports, which retain their previous states.
With a manual reset, since the on-chip supporting modules are initialized, ports used as on-chip supporting module I/O pins are switched to I/O ports controlled by DDR and DR.

**Table 4.3 Reset Types**

| | Reset Transition Conditions | | Internal State | |
|---|---|---|---|---|
| **Type** | **NMI** | **$\overline{\text{RES}}$** | **CPU** | **On-Chip Supporting Modules** |
| Power-on reset | High | Low | Initialized | Initialized |
| Manual reset | Low | Low | Initialized | Initialized, except for bus controller and I/O ports |

A reset caused by the watchdog timer can also be of either of two types: a power-on reset or a manual reset.

### 4.2.3 Reset Sequence

The H8S/2345 Series enters the reset state when the $\overline{\text{RES}}$ pin goes low.

To ensure that the H8S/2345 Series is reset, hold the $\overline{\text{RES}}$ pin low for at least 20 ms at power-up. To reset the H8S/2345 Series during operation, hold the $\overline{\text{RES}}$ pin low for at least 20 states.

When the $\overline{\text{RES}}$ pin goes high after being held low for the necessary time, the H8S/2345 Series starts reset exception handling as follows:

1. The internal state of the CPU and the registers of the on-chip supporting modules are initialized, the T bit is cleared to 0 in EXR, and the I bit is set to 1 in EXR and CCR.
2. The reset exception handling vector address is read and transferred to the PC, and program execution starts from the address indicated by the PC.

Figures 4.2 and 4.3 show examples of the reset sequence.

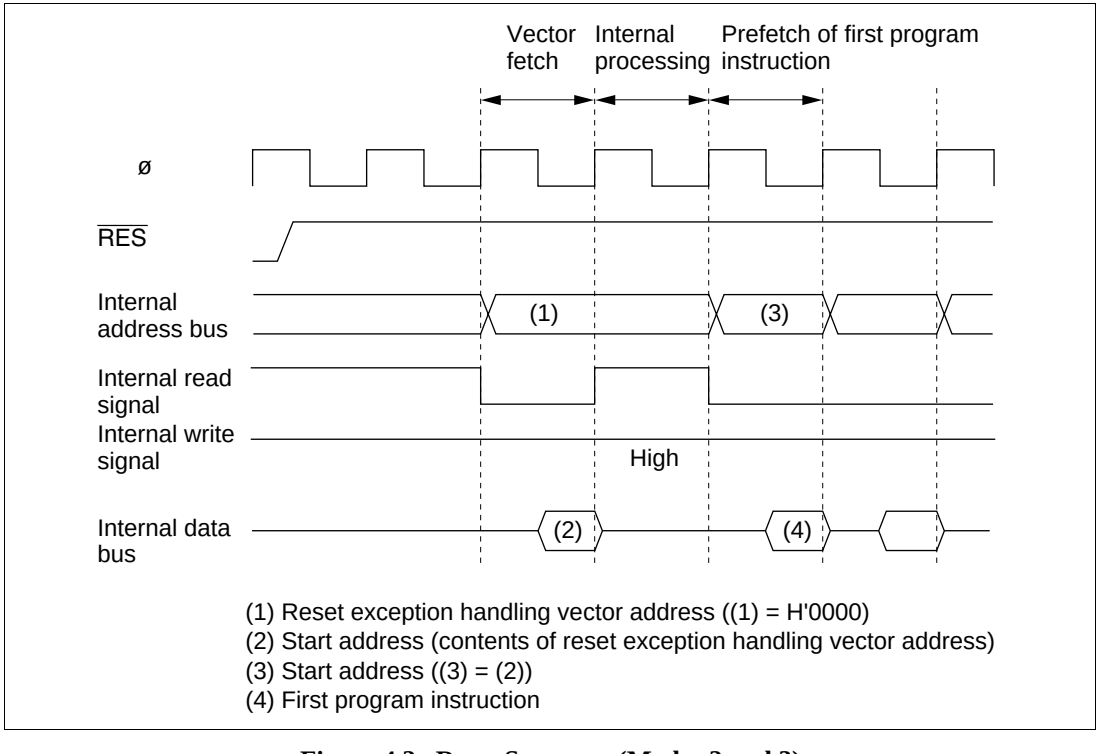
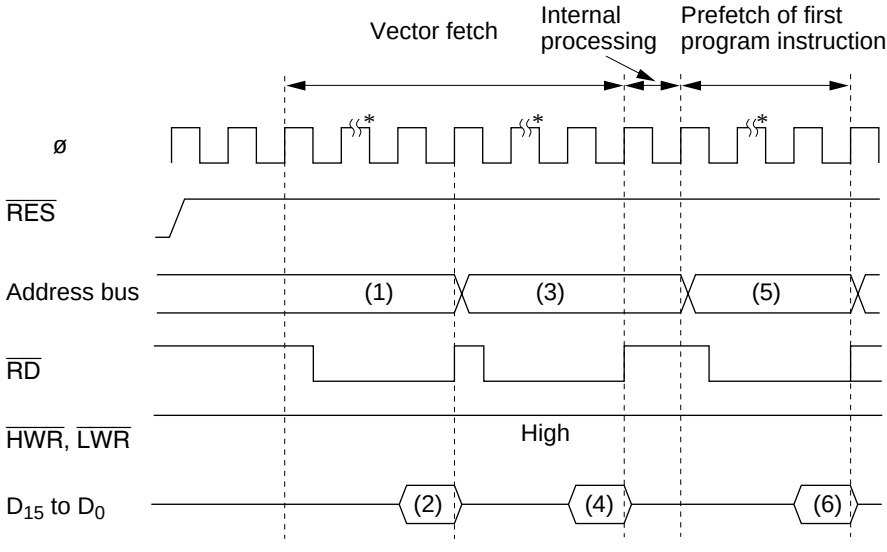



**Figure 4.2 Reset Sequence (Modes 2 and 3)**




(1) (3) Reset exception handling vector address ((1) = H'000000, (3) = H'000002)
(2) (4) Start address (contents of reset exception handling vector address)
(5) Start address ((5) = (2) (4))
(6) First program instruction

Note: * 3 program wait states are inserted.

**Figure 4.3 Reset Sequence (Mode 4)**

### 4.2.4 Interrupts after Reset

If an interrupt is accepted after a reset but before the stack pointer (SP) is initialized, the PC and CCR will not be saved correctly, leading to a program crash. To prevent this, all interrupt requests, including NMI, are disabled immediately after a reset. Since the first instruction of a program is always executed immediately after the reset state ends, make sure that this instruction initializes the stack pointer (example: MOV.L #xx:32, SP).

### 4.2.5 State of On-Chip Supporting Modules after Reset Release

After reset release, MSTPCR is initialized to H'3FFF and all modules except the DTC enter module stop mode. Consequently, on-chip supporting module registers cannot be read or written to. Register reading and writing is enabled when module stop mode is exited.

## 4.3 Traces

Traces are enabled in interrupt control mode 2. Trace mode is not activated in interrupt control mode 0, irrespective of the state of the T bit. For details of interrupt control modes, see section 5, Interrupt Controller.

If the T bit in EXR is set to 1, trace mode is activated. In trace mode, a trace exception occurs on completion of each instruction.

Trace mode is canceled by clearing the T bit in EXR to 0. It is not affected by interrupt masking.

Table 4.4 shows the state of CCR and EXR after execution of trace exception handling.

Interrupts are accepted even within the trace exception handling routine.

The T bit saved on the stack retains its value of 1, and when control is returned from the trace exception handling routine by the RTE instruction, trace mode resumes.

Trace exception handling is not carried out after execution of the RTE instruction.

**Table 4.4 Status of CCR and EXR after Trace Exception Handling**

| | CCR | | EXR | |
|---|---|---|---|---|
| **Interrupt Control Mode** | **I** | **UI** | **I2 to I0** | **T** |
| 0 | Trace exception handling cannot be used. | | | |
| 2 | 1 | — | — | 0 |

**Legend**

1: Set to 1

0: Cleared to 0

—: Retains value prior to execution.

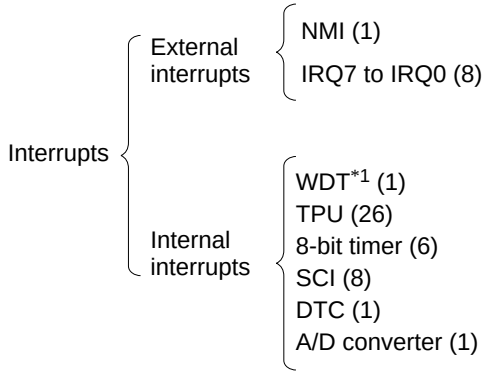
## 4.4 Interrupts

Interrupt exception handling can be requested by nine external sources (NMI, IRQ7 to IRQ0) and 43 internal sources in the on-chip supporting modules. Figure 4.4 classifies the interrupt sources and the number of interrupts of each type.

The on-chip supporting modules that can request interrupts include the watchdog timer (WDT), 16-bit timer-pulse unit (TPU), 8-bit timer, serial communication interface (SCI), data transfer controller (DTC), and A/D converter. Each interrupt source has a separate vector address.

NMI is the highest-priority interrupt. Interrupts are controlled by the interrupt controller. The interrupt controller has two interrupt control modes and can assign interrupts other than NMI to eight priority/mask levels to enable multiplexed interrupt control.

For details of interrupts, see section 5, Interrupt Controller.




Notes: Numbers in parentheses are the numbers of interrupt sources.
1. When the watchdog timer is used as an interval timer, it generates an interrupt request at each counter overflow.

**Figure 4.4 Interrupt Sources and Number of Interrupts**

## 4.5 Trap Instruction

Trap instruction exception handling starts when a TRAPA instruction is executed. Trap instruction exception handling can be executed at all times in the program execution state.

The TRAPA instruction fetches a start address from a vector table entry corresponding to a vector number from 0 to 3, as specified in the instruction code.

Table 4.5 shows the status of CCR and EXR after execution of trap instruction exception handling.

**Table 4.5 Status of CCR and EXR after Trap Instruction Exception Handling**

| | CCR | | EXR | |
|---|---|---|---|---|
| **Interrupt Control Mode** | **I** | **UI** | **I2 to I0** | **T** |
| 0 | 1 | — | — | — |
| 2 | 1 | — | — | 0 |

**Legend**

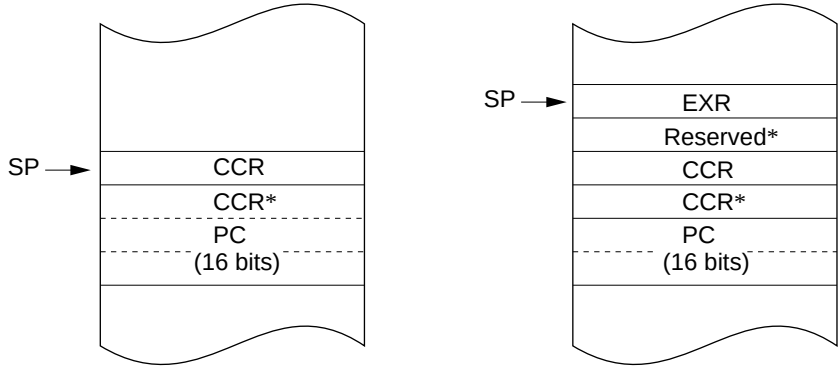
1: Set to 1

0: Cleared to 0

—: Retains value prior to execution.

## 4.6 Stack Status after Exception Handling

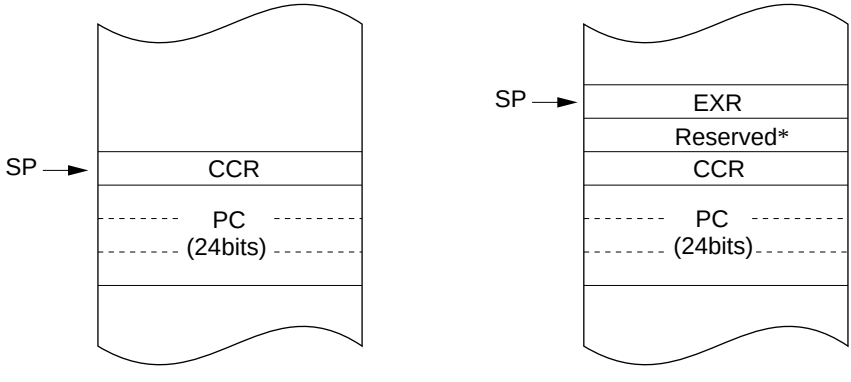
Figure 4.5 shows the stack after completion of trap instruction exception handling and interrupt exception handling.




(a) Interrupt control mode 0 (b) Interrupt control mode 2

Note: * Ignored on return.

**Figure 4.5 (1) Stack Status after Exception Handling (Normal Modes) (ZTAT, Mask ROM, and ROMless Versions Only)**




(a) Interrupt control mode 0 (b) Interrupt control mode 2

Note: * Ignored on return.

**Figure 4.5 (2) Stack Status after Exception Handling (Advanced Modes)**

## 4.7 Notes on Use of the Stack

When accessing word data or longword data, the H8S/2345 Series assumes that the lowest address bit is 0. The stack should always be accessed by word transfer instruction or longword transfer instruction, and the value of the stack pointer (SP, ER7) should always be kept even. Use the following instructions to save registers:

```
PUSH.W   Rn    (or MOV.W Rn, @-SP)
PUSH.L   ERn   (or MOV.L ERn, @-SP)
```

Use the following instructions to restore registers:

```
POP.W    Rn    (or MOV.W @SP+, Rn)
POP.L    ERn   (or MOV.L @SP+, ERn)
```

Setting SP to an odd value may lead to a malfunction. Figure 4.6 shows an example of what happens when the SP value is odd.

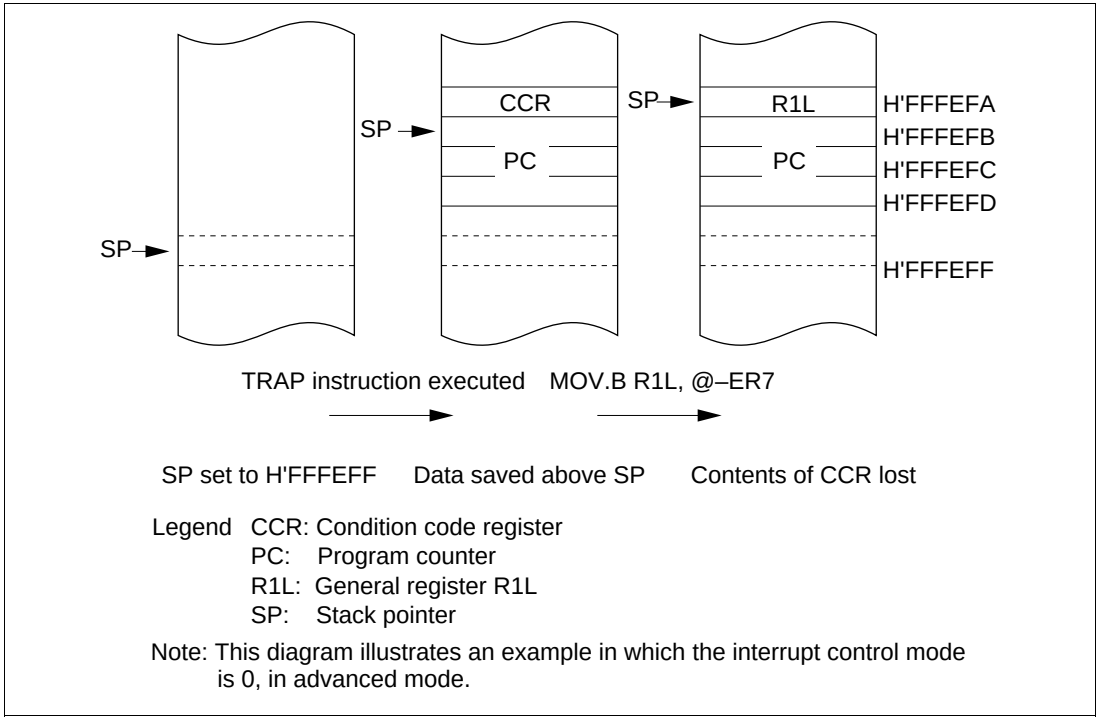



**Figure 4.6 Operation when SP Value is Odd**

# Section 5 Interrupt Controller

## 5.1 Overview

### 5.1.1 Features

The H8S/2345 Series controls interrupts by means of an interrupt controller. The interrupt controller has the following features:

- Two interrupt control modes
  - — Any of two interrupt control modes can be set by means of the INTM1 and INTM0 bits in the system control register (SYSCR).
- Priorities settable with IPR
  - — An interrupt priority register (IPR) is provided for setting interrupt priorities. Eight priority levels can be set for each module for all interrupts except NMI.
  - — NMI is assigned the highest priority level of 8, and can be accepted at all times.
- Independent vector addresses
  - — All interrupt sources are assigned independent vector addresses, making it unnecessary for the source to be identified in the interrupt handling routine.
- Nine external interrupts
  - — NMI is the highest-priority interrupt, and is accepted at all times. Rising edge or falling edge can be selected for NMI.
  - — Falling edge, rising edge, or both edge detection, or level sensing, can be selected for IRQ7 to IRQ0.
- DTC control
  - — DTC activation is performed by means of interrupts.

### 5.1.2 Block Diagram

A block diagram of the interrupt controller is shown in Figure 5.1.

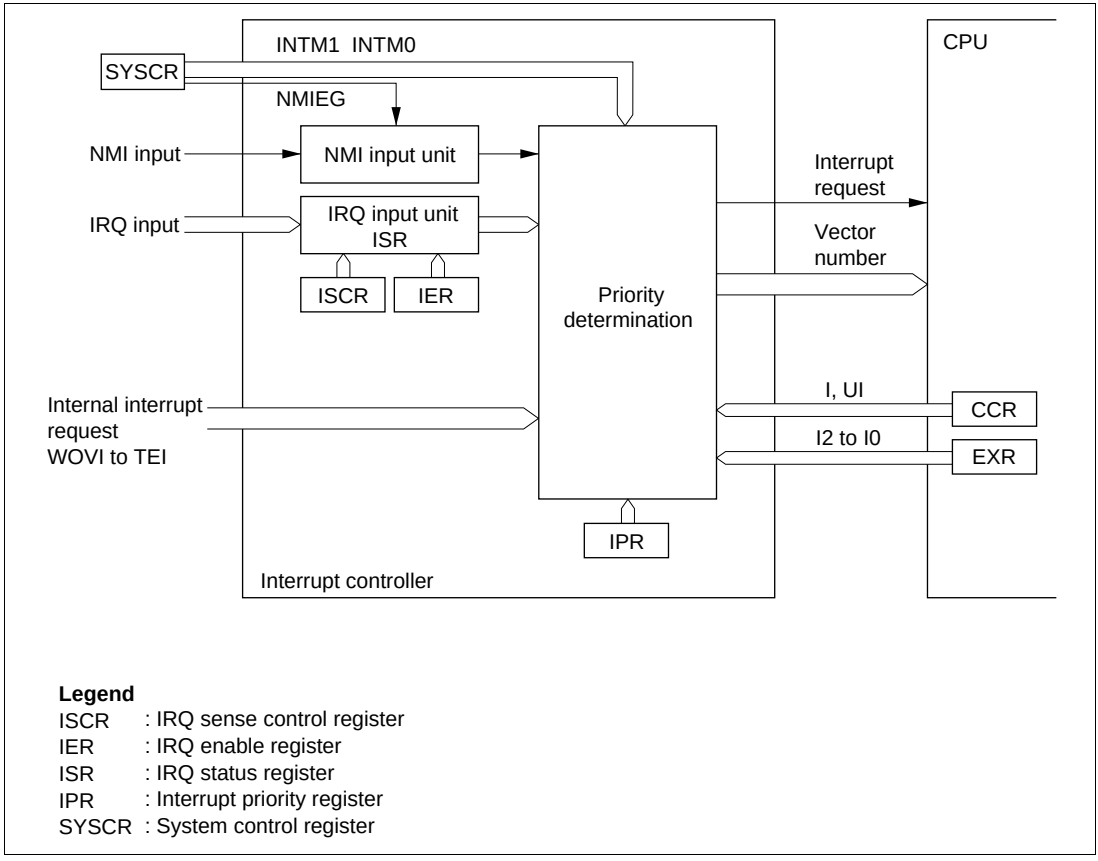



**Figure 5.1 Block Diagram of Interrupt Controller**

### 5.1.3 Pin Configuration

Table 5.1 summarizes the pins of the interrupt controller.

**Table 5.1 Interrupt Controller Pins**

| Name | Symbol | I/O | Function |
|---|---|---|---|
| Nonmaskable interrupt | NMI | Input | Nonmaskable external interrupt; rising or falling edge can be selected |
| External interrupt requests 7 to 0 | $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$ | Input | Maskable external interrupts; rising, falling, or both edges, or level sensing, can be selected |

### 5.1.4 Register Configuration

Table 5.2 summarizes the registers of the interrupt controller.

**Table 5.2 Interrupt Controller Registers**

| Name | Abbreviation | R/W | Initial Value | Address*[1] |
|---|---|---|---|---|
| System control register | SYSCR | R/W | H'01 | H'FF39 |
| IRQ sense control register H | ISCRH | R/W | H'00 | H'FF2C |
| IRQ sense control register L | ISCRL | R/W | H'00 | H'FF2D |
| IRQ enable register | IER | R/W | H'00 | H'FF2E |
| IRQ status register | ISR | R/(W)*[2] | H'00 | H'FF2F |
| Interrupt priority register A | IPRA | R/W | H'77 | H'FEC4 |
| Interrupt priority register B | IPRB | R/W | H'77 | H'FEC5 |
| Interrupt priority register C | IPRC | R/W | H'77 | H'FEC6 |
| Interrupt priority register D | IPRD | R/W | H'77 | H'FEC7 |
| Interrupt priority register E | IPRE | R/W | H'77 | H'FEC8 |
| Interrupt priority register F | IPRF | R/W | H'77 | H'FEC9 |
| Interrupt priority register G | IPRG | R/W | H'77 | H'FECA |
| Interrupt priority register H | IPRH | R/W | H'77 | H'FECB |
| Interrupt priority register I | IPRI | R/W | H'77 | H'FECC |
| Interrupt priority register J | IPRJ | R/W | H'77 | H'FECD |
| Interrupt priority register K | IPRK | R/W | H'77 | H'FECE |

Notes: 1. Lower 16 bits of the address.
2. Can only be written with 0 for flag clearing.

## 5.2 Register Descriptions

### 5.2.1 System Control Register (SYSCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | INTM1 | INTM0 | NMIEG | — | — | RAME |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

SYSCR is an 8-bit readable/writable register that selects the interrupt control mode, and the detected edge for NMI.

Only bits 5 to 3 are described here; for details of the other bits, see section 3.2.2, System Control Register (SYSCR).

SYSCR is initialized to H'01 by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bits 5 and 4—Interrupt Control Mode 1 and 0 (INTM1, INTM0):** These bits select one of two interrupt control modes for the interrupt controller.

| Bit 5 | Bit 4 | Interrupt | | |
|---|---|---|---|---|
| **INTM1** | **INTM0** | **Control Mode** | **Description** | |
| 0 | 0 | 0 | Interrupts are controlled by I bit | (Initial value) |
| | 1 | — | Setting prohibited | |
| 1 | 0 | 2 | Interrupts are controlled by bits I2 to I0, and IPR | |
| | 1 | — | Setting prohibited | |

**Bit 3—NMI Edge Select (NMIEG):** Selects the input edge for the NMI pin.

| Bit 3 | | |
|---|---|---|
| **NMIEG** | **Description** | |
| 0 | Interrupt request generated at falling edge of NMI input | (Initial value) |
| 1 | Interrupt request generated at rising edge of NMI input | |

### 5.2.2 Interrupt Priority Registers A to K (IPRA to IPRK)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 |
| Initial value: | | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| R/W | : | — | R/W | R/W | R/W | — | R/W | R/W | R/W |

The IPR registers are eleven 8-bit readable/writable registers that set priorities (levels 7 to 0) for interrupts other than NMI.

The correspondence between IPR settings and interrupt sources is shown in table 5.3.

The IPR registers set a priority (level 7 to 0) for each interrupt source other than NMI.

The IPR registers are initialized to H'77 by a reset and in hardware standby mode.

**Bits 7 and 3—Reserved:** Read-only bits, always read as 0.

**Table 5.3 Correspondence between Interrupt Sources and IPR Settings**

| | Bits | |
|---|---|---|
| **Register** | **6 to 4** | **2 to 0** |
| IPRA | IRQ0 | IRQ1 |
| IPRB | IRQ2<br>IRQ3 | IRQ4<br>IRQ5 |
| IPRC | IRQ6<br>IRQ7 | DTC |
| IPRD | Watchdog timer | —* |
| IPRE | —* | A/D converter |
| IPRF | TPU channel 0 | TPU channel 1 |
| IPRG | TPU channel 2 | TPU channel 3 |
| IPRH | TPU channel 4 | TPU channel 5 |
| IPRI | 8-bit timer channel 0 | 8-bit timer channel 1 |
| IPRJ | —* | SCI channel 0 |
| IPRK | SCI channel 1 | —* |

Note: * Reserved bits. May be read or written, but the setting is ignored.

As shown in table 5.3, multiple interrupts are assigned to one IPR. Setting a value in the range from H'0 to H'7 in the 3-bit groups of bits 6 to 4 and 2 to 0 sets the priority of the corresponding interrupt. The lowest priority level, level 0, is assigned by setting H'0, and the highest priority level, level 7, by setting H'7.

When interrupt requests are generated, the highest-priority interrupt according to the priority levels set in the IPR registers is selected. This interrupt level is then compared with the interrupt mask level set by the interrupt mask bits (I2 to I0) in the extend register (EXR) in the CPU, and if the priority level of the interrupt is higher than the set mask level, an interrupt request is issued to the CPU.

### 5.2.3 IRQ Enable Register (IER)

IER is an 8-bit readable/writable register that controls enabling and disabling of interrupt requests IRQ7 to IRQ0.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | IRQ7E | IRQ6E | IRQ5E | IRQ4E | IRQ3E | IRQ2E | IRQ1E | IRQ0E |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

IER is initialized to H'00 by a reset and in hardware standby mode.

**Bits 7 to 0—IRQ7 to IRQ0 Enable (IRQ7E to IRQ0E):** These bits select whether IRQ7 to IRQ0 are enabled or disabled.

| Bit n | | |
|---|---|---|
| **IRQnE** | **Description** | |
| 0 | IRQn interrupts disabled | (Initial value) |
| 1 | IRQn interrupts enabled | |

(n = 7 to 0)

### 5.2.4 IRQ Sense Control Registers H and L (ISCRH, ISCRL)

**ISCRH**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| | | IRQ7SCB | IRQ7SCA | IRQ6SCB | IRQ6SCA | IRQ5SCB | IRQ5SCA | IRQ4SCB | IRQ4SCA |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**ISCRL**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | IRQ3SCB | IRQ3SCA | IRQ2SCB | IRQ2SCA | IRQ1SCB | IRQ1SCA | IRQ0SCB | IRQ0SCA |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The ISCR registers are 16-bit readable/writable registers that select rising edge, falling edge, or both edge detection, or level sensing, for the input at pins $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$.

The ISCR registers are initialized to H'0000 by a reset and in hardware standby mode.

**Bits 15 to 0:** IRQ7 Sense Control A and B (IRQ7SCA, IRQ7SCB) to IRQ0 Sense Control A and B (IRQ0SCA, IRQ0SCB)

**Bits 15 to 0**

| IRQ7SCB to IRQ0SCB | IRQ7SCA to IRQ0SCA | Description |
|---|---|---|
| 0 | 0 | Interrupt request generated at $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$ input low level (Initial value) |
| | 1 | Interrupt request generated at falling edge of $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$ input |
| 1 | 0 | Interrupt request generated at rising edge of $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$ input |
| | 1 | Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$ input |

### 5.2.5 IRQ Status Register (ISR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | IRQ7F | IRQ6F | IRQ5F | IRQ4F | IRQ3F | IRQ2F | IRQ1F | IRQ0F |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* |

Note: * Only 0 can be written, to clear the flag.

ISR is an 8-bit readable/writable register that indicates the status of IRQ7 to IRQ0 interrupt requests.

ISR is initialized to H'00 by a reset and in hardware standby mode.

**Bits 7 to 0—IRQ7 to IRQ0 flags (IRQ7F to IRQ0F):** These bits indicate the status of IRQ7 to IRQ0 interrupt requests.

| Bit n<br>IRQnF | Description |
|---|---|
| 0 | [Clearing conditions] (Initial value)<br>• Cleared by reading IRQnF flag when IRQnF = 1, then writing 0 to IRQnF flag<br>• When interrupt exception handling is executed when low-level detection is set (IRQnSCB = IRQnSCA = 0) and $\overline{\text{IRQn}}$ input is high<br>• When IRQn interrupt exception handling is executed when falling, rising, or both-edge detection is set (IRQnSCB = 1 or IRQnSCA = 1)<br>• When the DTC is activated by an IRQn interrupt, and the DISEL bit in MRB of the DTC is cleared to 0 |
| 1 | [Setting conditions]<br>• When $\overline{\text{IRQn}}$ input goes low when low-level detection is set (IRQnSCB = IRQnSCA = 0)<br>• When a falling edge occurs in $\overline{\text{IRQn}}$ input when falling edge detection is set (IRQnSCB = 0, IRQnSCA = 1)<br>• When a rising edge occurs in $\overline{\text{IRQn}}$ input when rising edge detection is set (IRQnSCB = 1, IRQnSCA = 0)<br>• When a falling or rising edge occurs in $\overline{\text{IRQn}}$ input when both-edge detection is set (IRQnSCB = IRQnSCA = 1) |

(n = 7 to 0)

## 5.3 Interrupt Sources

Interrupt sources comprise external interrupts (NMI and IRQ7 to IRQ0) and internal interrupts (43 sources).

### 5.3.1 External Interrupts

There are nine external interrupts: NMI and IRQ7 to IRQ0. Of these, NMI and IRQ2 to IRQ0 can be used to restore the H8S/2345 Series from software standby mode.

**NMI Interrupt:** NMI is the highest-priority interrupt, and is always accepted by the CPU regardless of the status of the CPU interrupt mask bits. The NMIEG bit in SYSCR can be used to select whether an interrupt is requested at a rising edge or a falling edge on the NMI pin.

The vector number for NMI interrupt exception handling is 7.

**IRQ7 to IRQ0 Interrupts:** Interrupts IRQ7 to IRQ0 are requested by an input signal at pins $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$. Interrupts IRQ7 to IRQ0 have the following features:

- Using ISCR, it is possible to select whether an interrupt is generated by a low level, falling edge, rising edge, or both edges, at pins $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$.
- Enabling or disabling of interrupt requests IRQ7 to IRQ0 can be selected with IER.
- The interrupt priority level can be set with IPR.
- The status of interrupt requests IRQ7 to IRQ0 is indicated in ISR. ISR flags can be cleared to 0 by software.

A block diagram of interrupts IRQ7 to IRQ0 is shown in figure 5.2.

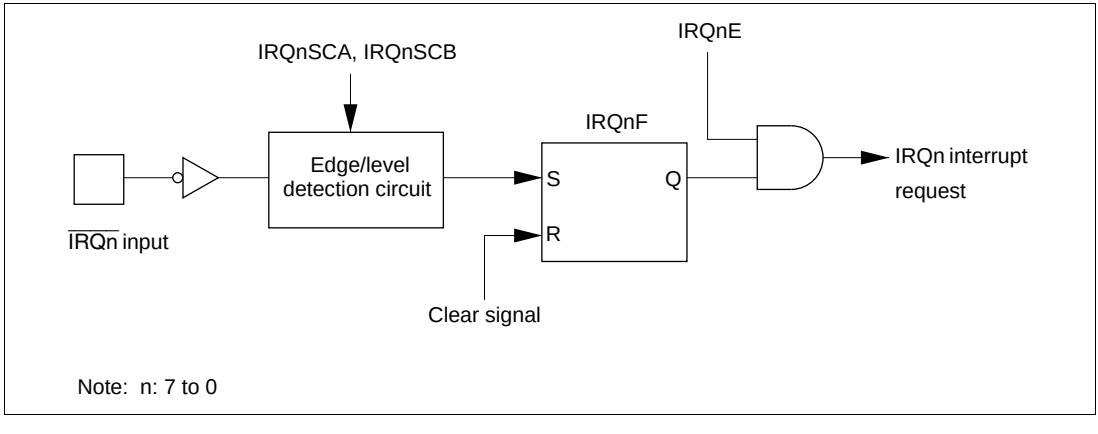



**Figure 5.2 Block Diagram of Interrupts IRQ7 to IRQ0**

Figure 5.3 shows the timing of setting IRQnF.

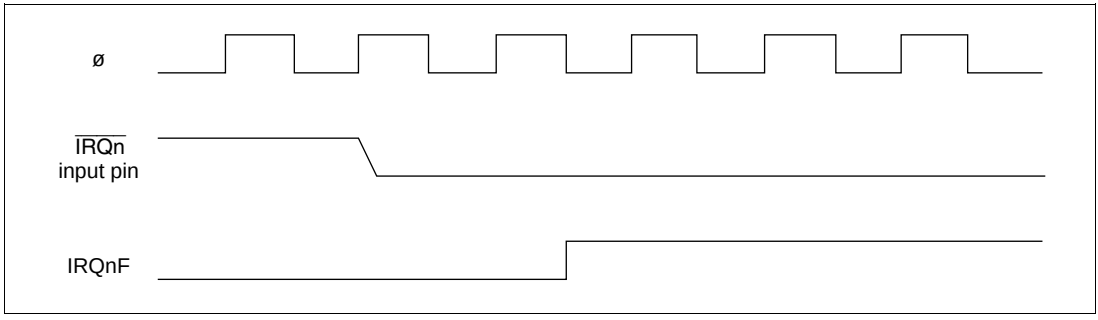



**Figure 5.3 Timing of Setting IRQnF**

The vector numbers for IRQ7 to IRQ0 interrupt exception handling are 23 to 16.

Detection of IRQ7 to IRQ0 interrupts does not depend on whether the relevant pin has been set for input or output. However, when a pin is used as an external interrupt input pin, do not clear the corresponding DDR to 0 and use the pin as an I/O pin for another function.

### 5.3.2 Internal Interrupts

There are 43 sources for internal interrupts from on-chip supporting modules.

- For each on-chip supporting module there are flags that indicate the interrupt request status, and enable bits that select enabling or disabling of these interrupts. If both of these are set to 1 for a particular interrupt source, an interrupt request is issued to the interrupt controller.
- The interrupt priority level can be set by means of IPR.
- The DTC can be activated by a TPU, 8-bit timer, SCI, or other interrupt request. When the DTC is activated by an interrupt, the interrupt control mode and interrupt mask bits are not affected.

### 5.3.3 Interrupt Exception Handling Vector Table

Table 5.4 shows interrupt exception handling sources, vector addresses, and interrupt priorities. For default priorities, the lower the vector number, the higher the priority.

Priorities among modules can be set by means of the IPR. The situation when two or more modules are set to the same priority, and priorities within a module, are fixed as shown in table 5.4.

**Table 5.4 Interrupt Sources, Vector Addresses, and Interrupt Priorities**

| Interrupt Source | Origin of Interrupt Source | Vector Number | Vector Address*1 Normal Mode*2 | Vector Address*1 Advanced Mode | IPR | Priority |
|---|---|---|---|---|---|---|
| NMI | External pin | 7 | H'000E | H'001C | | High |
| IRQ0 | | 16 | H'0020 | H'0040 | IPRA6 to 4 | |
| IRQ1 | | 17 | H'0022 | H'0044 | IPRA2 to 0 | |
| IRQ2 | | 18 | H'0024 | H'0048 | IPRB6 to 4 | |
| IRQ3 | | 19 | H'0026 | H'004C | | |
| IRQ4 | | 20 | H'0028 | H'0050 | IPRB2 to 0 | |
| IRQ5 | | 21 | H'002A | H'0054 | | |
| IRQ6 | | 22 | H'002C | H'0058 | IPRC6 to 4 | |
| IRQ7 | | 23 | H'002E | H'005C | | |
| SWDTEND (software activation interrupt end) | DTC | 24 | H'0030 | H'0060 | IPRC2 to 0 | |
| WOVI (interval timer) | Watchdog timer | 25 | H'0032 | H'0064 | IPRD6 to 4 | |
| Reserved | — | 26 | H'0034 | H'0068 | | |
| | | 27 | H'0036 | H'006C | | |
| ADI (A/D conversion end) | A/D | 28 | H'0038 | H'0070 | IPRE2 to 0 | |
| Reserved | — | 29 | H'003A | H'0074 | | |
| | | 30 | H'003C | H'0078 | | |
| | | 31 | H'003E | H'007C | | |
| TGI0A (TGR0A input capture/compare match) | TPU channel 0 | 32 | H'0040 | H'0080 | IPRF6 to 4 | |
| TGI0B (TGR0B input capture/compare match) | | 33 | H'0042 | H'0084 | | |
| TGI0C (TGR0C input capture/compare match) | | 34 | H'0044 | H'0088 | | |
| TGI0D (TGR0D input capture/compare match) | | 35 | H'0046 | H'008C | | |
| TCI0V (overflow 0) | | 36 | H'0048 | H'0090 | | |
| Reserved | — | 37 | H'004A | H'0094 | | |
| | | 38 | H'004C | H'0098 | | |
| | | 39 | H'004E | H'009C | | Low |

Notes: 1. Lower 16 bits of the start address.
2. ZTAT, mask ROM, and ROMless versions only.

**Table 5.4 Interrupt Sources, Vector Addresses, and Interrupt Priorities (cont)**

| Interrupt Source | Origin of Interrupt Source | Vector Number | Vector Address*1 Normal Mode*2 | Vector Address*1 Advanced Mode | IPR | Priority |
|---|---|---|---|---|---|---|
| TGI1A (TGR1A input capture/compare match) | TPU channel 1 | 40 | H'0050 | H'00A0 | IPRF2 to 0 | High |
| TGI1B (TGR1B input capture/compare match) | | 41 | H'0052 | H'00A4 | | |
| TCI1V (overflow 1) | | 42 | H'0054 | H'00A8 | | |
| TCI1U (underflow 1) | | 43 | H'0056 | H'00AC | | |
| TGI2A (TGR2A input capture/compare match) | TPU channel 2 | 44 | H'0058 | H'00B0 | IPRG6 to 4 | |
| TGI2B (TGR2B input capture/compare match) | | 45 | H'005A | H'00B4 | | |
| TCI2V (overflow 2) | | 46 | H'005C | H'00B8 | | |
| TCI2U (underflow 2) | | 47 | H'005E | H'00BC | | |
| TGI3A (TGR3A input capture/compare match) | TPU channel 3 | 48 | H'0060 | H'00C0 | IPRG2 to 0 | |
| TGI3B (TGR3B input capture/compare match) | | 49 | H'0062 | H'00C4 | | |
| TGI3C (TGR3C input capture/compare match) | | 50 | H'0064 | H'00C8 | | |
| TGI3D (TGR3D input capture/compare match) | | 51 | H'0066 | H'00CC | | |
| TCI3V (overflow 1) | | 52 | H'0068 | H'00D0 | | |
| Reserved | — | 53 | H'006A | H'00D4 | | |
| | | 54 | H'006C | H'00D8 | | |
| | | 55 | H'006E | H'00DC | | |
| TGI4A (TGR4A input capture/compare match) | TPU channel 4 | 56 | H'0070 | H'00E0 | IPRH6 to 4 | |
| TGI4B (TGR4B input capture/compare match) | | 57 | H'0072 | H'00E4 | | |
| TCI4V (overflow 4) | | 58 | H'0074 | H'00E8 | | |
| TCI4U (underflow 4) | | 59 | H'0076 | H'00EC | | |
| TGI5A (TGR5A input capture/compare match) | TPU channel 5 | 60 | H'0078 | H'00F0 | IPRH2 to 0 | |
| TGI5B (TGR5B input capture/compare match) | | 61 | H'007A | H'00F4 | | |
| TCI5V (overflow 5) | | 62 | H'007C | H'00F8 | | |
| TCI5U (underflow 5) | | 63 | H'007E | H'00FC | | Low |

Notes: 1. Lower 16 bits of the start address.
2. ZTAT, mask ROM, and ROMless versions only.

**Table 5.4 Interrupt Sources, Vector Addresses, and Interrupt Priorities (cont)**

| Interrupt Source | Origin of Interrupt Source | Vector Number | Vector Address*1 Normal Mode*2 | Vector Address*1 Advanced Mode | IPR | Priority |
|---|---|---|---|---|---|---|
| CMIA0 (compare match A0) | 8-bit timer | 64 | H'0080 | H'0100 | IPRI6 to 4 | High |
| CMIB0 (compare match B0) | channel 0 | 65 | H'0082 | H'0104 | | |
| OVI0 (overflow 0) | | 66 | H'0084 | H'0108 | | |
| Reserved | — | 67 | H'0086 | H'010C | | |
| CMIA1 (compare match A1) | 8-bit timer | 68 | H'0088 | H'0110 | IPRI2 to 0 | |
| CMIB1 (compare match B1) | channel 1 | 69 | H'008A | H'0114 | | |
| OVI1 (overflow 1) | | 70 | H'008C | H'0118 | | |
| Reserved | — | 71 | H'008E | H'011C | | |
| | | 72 | H'0090 | H'0120 | | |
| | | 73 | H'0092 | H'0124 | | |
| | | 74 | H'0094 | H'0128 | | |
| | | 75 | H'0096 | H'012C | | |
| | | 76 | H'0098 | H'0130 | | |
| | | 77 | H'009A | H'0134 | | |
| | | 78 | H'009C | H'0138 | | |
| | | 79 | H'009E | H'013C | | |
| ERI0 (receive error 0) | SCI | 80 | H'00A0 | H'0140 | IPRJ2 to 0 | |
| RXI0 (reception completed 0) | channel 0 | 81 | H'00A2 | H'0144 | | |
| TXI0 (transmit data empty 0) | | 82 | H'00A4 | H'0148 | | |
| TEI0 (transmission end 0) | | 83 | H'00A6 | H'014C | | |
| ERI1 (receive error 1) | SCI | 84 | H'00A8 | H'0150 | IPRK6 to 4 | |
| RXI1 (reception completed 1) | channel 1 | 85 | H'00AA | H'0154 | | |
| TXI1 (transmit data empty 1) | | 86 | H'00AC | H'0158 | | |
| TEI1 (transmission end 1) | | 87 | H'00AE | H'015C | | Low |

Notes: 1. Lower 16 bits of the start address.

2. ZTAT, mask ROM, and ROMless versions only.

## 5.4 Interrupt Operation

### 5.4.1 Interrupt Control Modes and Interrupt Operation

Interrupt operations in the H8S/2345 Series differ depending on the interrupt control mode.

NMI interrupts are accepted at all times except in the reset state and the hardware standby state. In the case of IRQ interrupts and on-chip supporting module interrupts, an enable bit is provided for each interrupt. Clearing an enable bit to 0 disables the corresponding interrupt request. Interrupt sources for which the enable bits are set to 1 are controlled by the interrupt controller.

Table 5.5 shows the interrupt control modes.

The interrupt controller performs interrupt control according to the interrupt control mode set by the INTM1 and INTM0 bits in SYSCR, the priorities set in IPR, and the masking state indicated by the I and UI bits in the CPU's CCR, and bits I2 to I0 in EXR.

**Table 5.5 Interrupt Control Modes**

| Interrupt Control Mode | SYSCR | | Priority Setting Registers | Interrupt Mask Bits | Description |
|---|---|---|---|---|---|
| | INTM1 | INTM0 | | | |
| 0 | 0 | 0 | — | I | Interrupt mask control is performed by the I bit. |
| — | | 1 | — | — | Setting prohibited |
| 2 | 1 | 0 | IPR | I2 to I0 | 8-level interrupt mask control is performed by bits I2 to I0. 8 priority levels can be set with IPR. |
| — | | 1 | — | — | Setting prohibited |

Figure 5.4 shows a block diagram of the priority decision circuit.

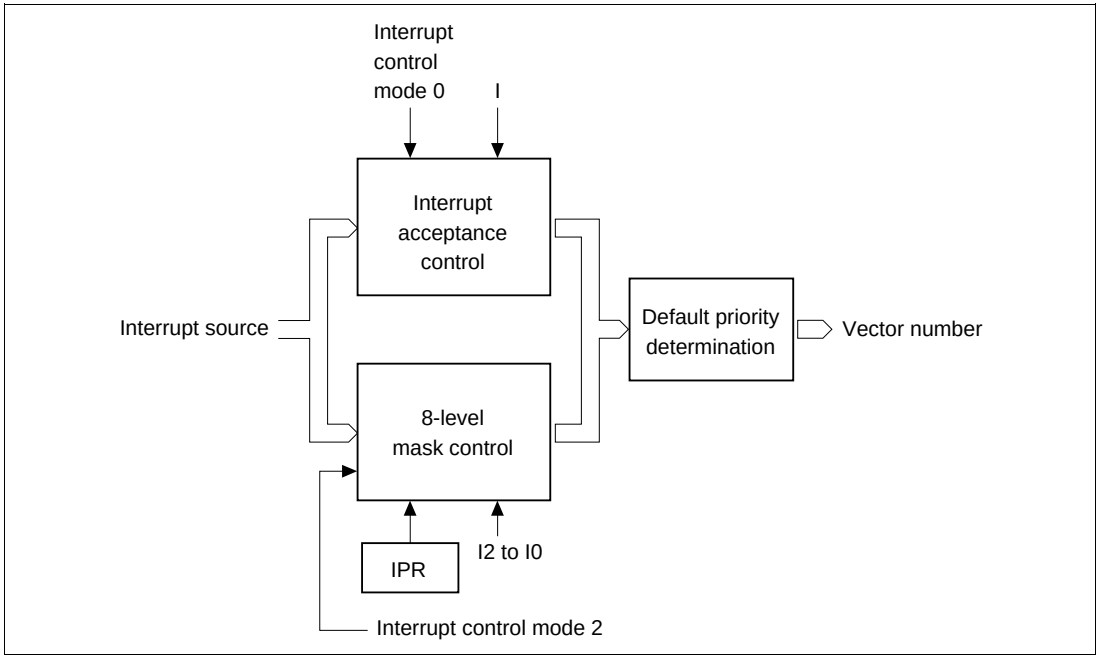



**Figure 5.4 Block Diagram of Interrupt Control Operation**

### (1) Interrupt Acceptance Control

In interrupt control mode 0, interrupt acceptance is controlled by the I bit in CCR.

Table 5.6 shows the interrupts selected in each interrupt control mode.

**Table 5.6 Interrupts Selected in Each Interrupt Control Mode (1)**

| | Interrupt Mask Bits | |
|---|---|---|
| **Interrupt Control Mode** | **I** | **Selected Interrupts** |
| 0 | 0 | All interrupts |
| | 1 | NMI interrupts |
| 2 | * | All interrupts |

**Legend**
* : Don't care

### (2) 8-Level Control

In interrupt control mode 2, 8-level mask level determination is performed for the selected interrupts in interrupt acceptance control according to the interrupt priority level (IPR).

The interrupt source selected is the interrupt with the highest priority level, and whose priority level set in IPR is higher than the mask level.

**Table 5.7 Interrupts Selected in Each Interrupt Control Mode (2)**

| Interrupt Control Mode | Selected Interrupts |
|---|---|
| 0 | All interrupts |
| 2 | Highest-priority-level (IPR) interrupt whose priority level is greater than the mask level (IPR > I2 to I0). |

### (3) Default Priority Determination

When an interrupt is selected by 8-level control, its priority is determined and a vector number is generated.

If the same value is set for IPR, acceptance of multiple interrupts is enabled, and so only the interrupt source with the highest priority according to the preset default priorities is selected and has a vector number generated.

Interrupt sources with a lower priority than the accepted interrupt source are held pending.

Table 5.8 shows operations and control signal functions in each interrupt control mode.

**Table 5.8 Operations and Control Signal Functions in Each Interrupt Control Mode**

| Interrupt Control Mode | Setting | | Interrupt Acceptance Control | | | 8-Level Control | | Default Priority Determination | T (Trace) |
|---|---|---|---|---|---|---|---|---|---|
| | INTM1 | INTM0 | I | | | I2 to I0 | IPR | | |
| 0 | 0 | 0 | ○ | IM | X | — | —*2 | ○ | — |
| 2 | 1 | 0 | X | —*1 | ○ | IM | PR | ○ | T |

**Legend**

○ : Interrupt operation control performed
X : No operation. (All interrupts enabled)
IM : Used as interrupt mask bit
PR : Sets priority.
— : Not used.
*1 : Set to 1 when interrupt is accepted.
*2 : Keep the initial setting.

### 5.4.2 Interrupt Control Mode 0

Enabling and disabling of IRQ interrupts and on-chip supporting module interrupts can be set by means of the I bit in the CPU's CCR. Interrupts are enabled when the I bit is cleared to 0, and disabled when set to 1.

Figure 5.5 shows a flowchart of the interrupt acceptance operation in this case.

[1] If an interrupt source occurs when the corresponding interrupt enable bit is set to 1, an interrupt request is sent to the interrupt controller.

[2] The I bit is then referenced. If the I bit is cleared to 0, the interrupt request is accepted. If the I bit is set to 1, only an NMI interrupt is accepted, and other interrupt requests are held pending.

[3] Interrupt requests are sent to the interrupt controller, the highest-ranked interrupt according to the priority system is accepted, and other interrupt requests are held pending.

[4] When an interrupt request is accepted, interrupt exception handling starts after execution of the current instruction has been completed.

[5] The PC and CCR are saved to the stack area by interrupt exception handling. The PC saved on the stack shows the address of the first instruction to be executed after returning from the interrupt handling routine.

[6] Next, the I bit in CCR is set to 1. This masks all interrupts except NMI.

[7] A vector address is generated for the accepted interrupt, and execution of the interrupt handling routine starts at the address indicated by the contents of that vector address.

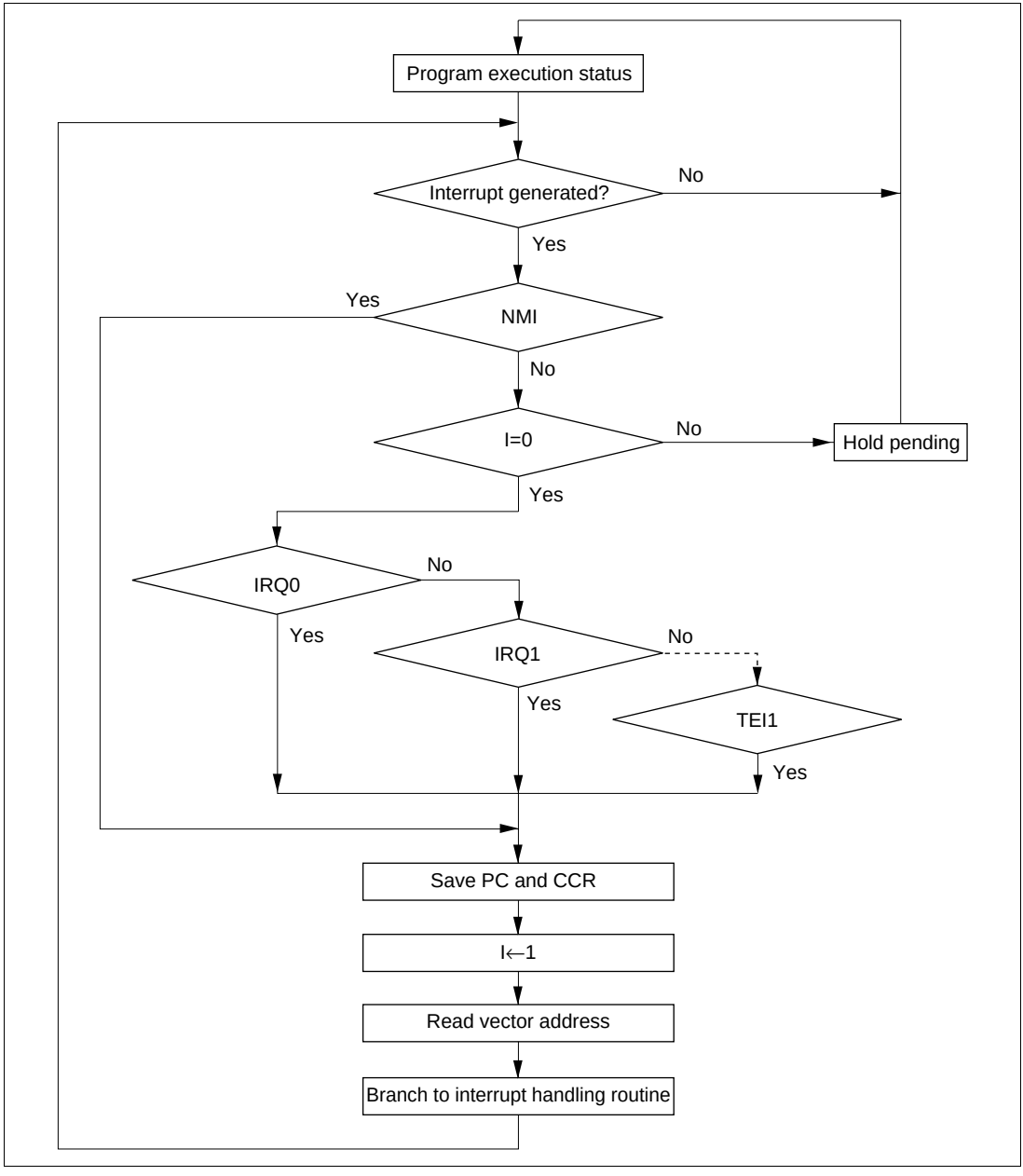



**Figure 5.5 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 0**

### 5.4.3 Interrupt Control Mode 2

Eight-level masking is implemented for IRQ interrupts and on-chip supporting module interrupts by comparing the interrupt mask level set by bits I2 to I0 of EXR in the CPU with IPR.

Figure 5.6 shows a flowchart of the interrupt acceptance operation in this case.

[1] If an interrupt source occurs when the corresponding interrupt enable bit is set to 1, an interrupt request is sent to the interrupt controller.

[2] When interrupt requests are sent to the interrupt controller, the interrupt with the highest priority according to the interrupt priority levels set in IPR is selected, and lower-priority interrupt requests are held pending. If a number of interrupt requests with the same priority are generated at the same time, the interrupt request with the highest priority according to the priority system shown in table 5.4 is selected.

[3] Next, the priority of the selected interrupt request is compared with the interrupt mask level set in EXR. An interrupt request with a priority no higher than the mask level set at that time is held pending, and only an interrupt request with a priority higher than the interrupt mask level is accepted.

[4] When an interrupt request is accepted, interrupt exception handling starts after execution of the current instruction has been completed.

[5] The PC, CCR, and EXR are saved to the stack area by interrupt exception handling. The PC saved on the stack shows the address of the first instruction to be executed after returning from the interrupt handling routine.

[6] The T bit in EXR is cleared to 0. The interrupt mask level is rewritten with the priority level of the accepted interrupt.

If the accepted interrupt is NMI, the interrupt mask level is set to H'7.

[7] A vector address is generated for the accepted interrupt, and execution of the interrupt handling routine starts at the address indicated by the contents of that vector address.

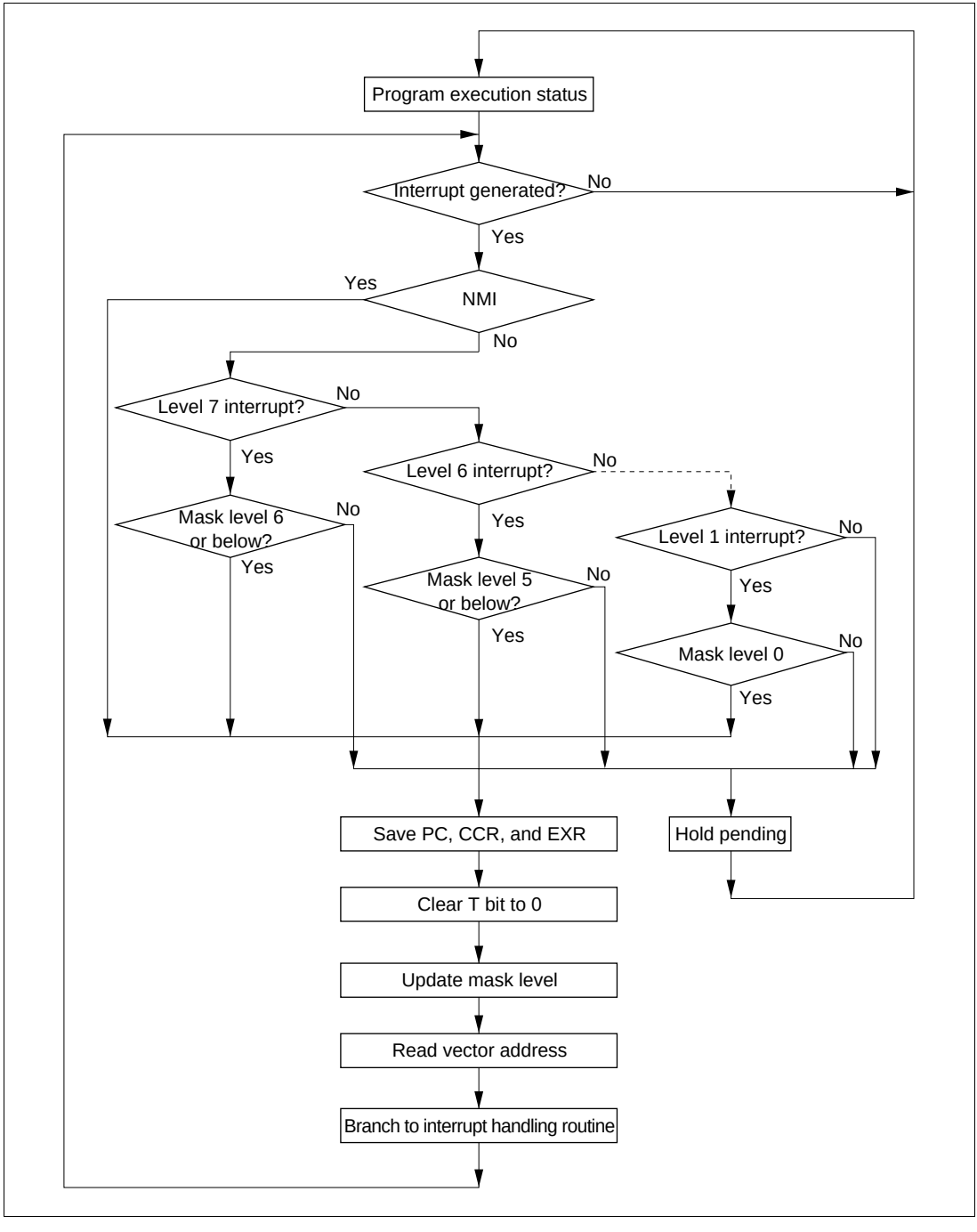
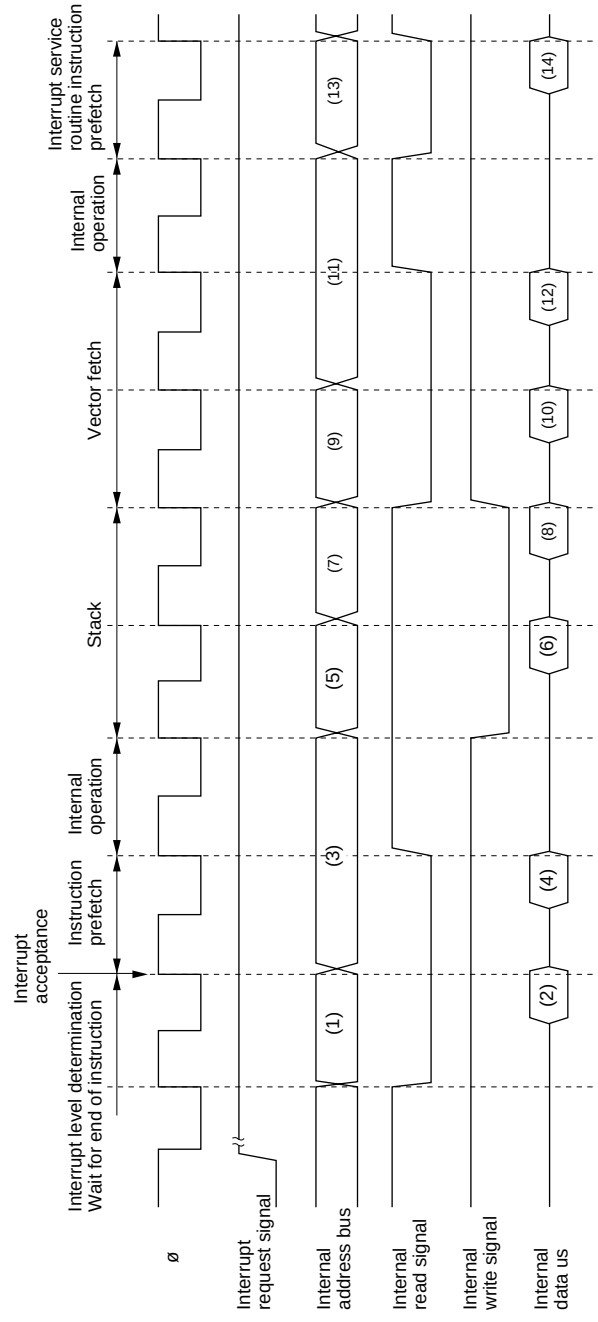



**Figure 5.6 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 2**

### 5.4.4 Interrupt Exception Handling Sequence

Figure 5.7 shows the interrupt exception handling sequence. The example shown is for the case where interrupt control mode 0 is set in advanced mode, and the program area and stack area are in on-chip memory.




(1) Instruction prefetch address (Not executed. This is the contents of the saved PC, the return address.)
(2) (4) Instruction code (Not executed.)
(3) Instruction prefetch address (Not executed.)
(5) SP-2
(7) SP-4
(6) (8) Saved PC and saved CCR
(9) (11) Vector address
(10) (12) Interrupt handling routine start address (vector address contents)
(13) Interrupt handling routine start address ((13) = (10) (12))
(14) First instruction of interrupt handling routine

**Figure 5.7 Interrupt Exception Handling**

### 5.4.5 Interrupt Response Times

The H8S/2345 Series is capable of fast word transfer instruction to on-chip memory, and the program area is provided in on-chip ROM and the stack area in on-chip RAM, enabling high-speed processing.

Table 5.9 shows interrupt response times - the interval between generation of an interrupt request and execution of the first instruction in the interrupt handling routine. The execution status symbols used in table 5.9 are explained in table 5.10.

**Table 5.9 Interrupt Response Times**

| | | Normal Mode*5 | | Advanced Mode | |
|---|---|---|---|---|---|
| **No.** | **Execution Status** | **INTM1 = 0** | **INTM1 = 1** | **INTM1 = 0** | **INTM1 = 1** |
| 1 | Interrupt priority determination*1 | 3 | 3 | 3 | 3 |
| 2 | Number of wait states until executing instruction ends*2 | 1 to $19+2\cdot S_I$ | 1 to $19+2\cdot S_I$ | 1 to $19+2\cdot S_I$ | 1 to $19+2\cdot S_I$ |
| 3 | PC, CCR, EXR stack save | $2\cdot S_K$ | $3\cdot S_K$ | $2\cdot S_K$ | $3\cdot S_K$ |
| 4 | Vector fetch | $S_I$ | $S_I$ | $2\cdot S_I$ | $2\cdot S_I$ |
| 5 | Instruction fetch*3 | $2\cdot S_I$ | $2\cdot S_I$ | $2\cdot S_I$ | $2\cdot S_I$ |
| 6 | Internal processing*4 | 2 | 2 | 2 | 2 |
| Total (using on-chip memory) | | 11 to 31 | 12 to 32 | 12 to 32 | 13 to 33 |

Notes: 1. Two states in case of internal interrupt.
2. Refers to MULXS and DIVXS instructions.
3. Prefetch after interrupt acceptance and interrupt handling routine prefetch.
4. Internal processing after interrupt acceptance and internal processing after vector fetch.
5. ZTAT, mask ROM, and ROMless versions only.

**Table 5.10 Number of States in Interrupt Handling Routine Execution Statuses**

| | | Object of Access | | | | |
|---|---|---|---|---|---|---|
| | | | External Device | | | |
| | | | 8 Bit Bus | | 16 Bit Bus | |
| **Symbol** | | **Internal Memory** | **2-State Access** | **3-State Access** | **2-State Access** | **3-State Access** |
| Instruction fetch | $S_I$ | 1 | 4 | 6+2m | 2 | 3+m |
| Branch address read | $S_J$ | | | | | |
| Stack manipulation | $S_K$ | | | | | |

**Legend**

m : Number of wait states in an external device access.

## 5.5 Usage Notes

### 5.5.1 Contention between Interrupt Generation and Disabling

When an interrupt enable bit is cleared to 0 to disable interrupts, the disabling becomes effective after execution of the instruction.

In other words, when an interrupt enable bit is cleared to 0 by an instruction such as BCLR or MOV, if an interrupt is generated during execution of the instruction, the interrupt concerned will still be enabled on completion of the instruction, and so interrupt exception handling for that interrupt will be executed on completion of the instruction. However, if there is an interrupt request of higher priority than that interrupt, interrupt exception handling will be executed for the higher-priority interrupt, and the lower-priority interrupt will be ignored.

The same also applies when an interrupt source flag is cleared.

Figure 5.8 shows and example in which the CMIEA bit in 8-bit timer TCR is cleared to 0.

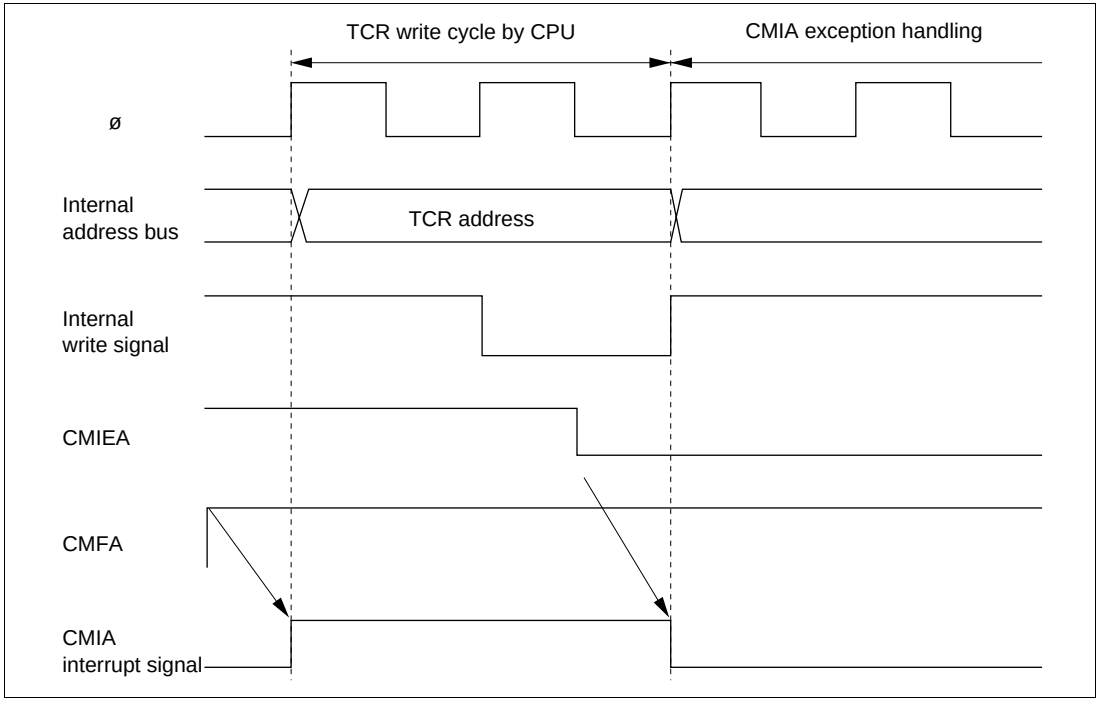



**Figure 5.8 Contention between Interrupt Generation and Disabling**

The above contention will not occur if an enable bit or interrupt source flag is cleared to 0 while the interrupt is masked.

### 5.5.2 Instructions that Disable Interrupts

Instructions that disable interrupts are LDC, ANDC, ORC, and XORC. After any of these instructions is executed, all interrupts including NMI are disabled and the next instruction is always executed. When the I bit is set by one of these instructions, the new value becomes valid two states after execution of the instruction ends.

### 5.5.3 Times when Interrupts are Disabled

There are times when interrupt acceptance is disabled by the interrupt controller.

The interrupt controller disables interrupt acceptance for a 3-state period after the CPU has updated the mask level with an LDC, ANDC, ORC, or XORC instruction.

### 5.5.4 Interrupts during Execution of EEPMOV Instruction

Interrupt operation differs between the EEPMOV.B instruction and the EEPMOV.W instruction.

With the EEPMOV.B instruction, an interrupt request (including NMI) issued during the transfer is not accepted until the move is completed.

With the EEPMOV.W instruction, if an interrupt request is issued during the transfer, interrupt exception handling starts at a break in the transfer cycle. The PC value saved on the stack in this case is the address of the next instruction.

Therefore, if an interrupt is generated during execution of an EEPMOV.W instruction, the following coding should be used.

```
L1:   EEPMOV.W
      MOV.W     R4,R4
      BNE       L1
```

## 5.6 DTC Activation by Interrupt

### 5.6.1 Overview

The DTC can be activated by an interrupt. In this case, the following options are available:

- Interrupt request to CPU
- Activation request to DTC
- Selection of a number of the above

For details of interrupt requests that can be used with to activate the DTC, see section 7, Data Transfer Controller.

### 5.6.2 Block Diagram

Figure 5.9 shows a block diagram of the DTC interrupt controller.

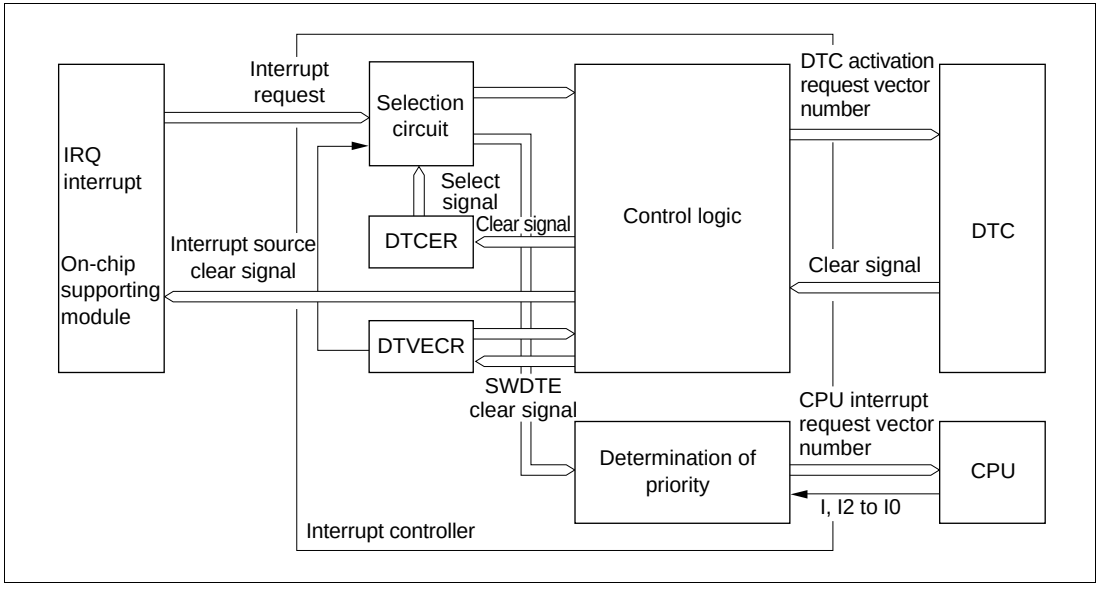



**Figure 5.9 Interrupt Control for DTC and DMAC**

### 5.6.3 Operation

The interrupt controller has three main functions in DTC control.

**(1) Selection of Interrupt Source:** Interrupt sources can be specified as DTC activation requests or CPU interrupt requests by means of the DTCE bit of DTCEA to DTCEE in the DTC.

After a DTC data transfer, the DTCE bit can be cleared to 0 and an interrupt request sent to the CPU in accordance with the specification of the DISEL bit of MRB in the DTC.

When the DTC has performed the specified number of data transfers and the transfer counter value is zero, the DTCE bit is cleared to 0 and an interrupt request is sent to the CPU after the DTC data transfer.

**(2) Determination of Priority:** The DTC activation source is selected in accordance with the default priority order, and is not affected by mask or priority levels. See section 7.3.3, DTC Vector Table, for the respective priorities.

**(3) Operation Order:** If the same interrupt is selected as a DTC activation source and a CPU interrupt source, the DTC data transfer is performed first, followed by CPU interrupt exception handling.

If the same interrupt is selected as a DTC activation source or CPU interrupt source, operations are performed for them independently according to their respective operating statuses and bus mastership priorities.

Table 5.11 summarizes interrupt source selection and interrupt source clearance control according to the settings of the DTCE bit of DTCEA to DTCEE in the DTC and the DISEL bit of MRB in the DTC.

**Table 5.11 Interrupt Source Selection and Clearing Control**

| Settings | | | |
|---|---|---|---|
| **DTC** | | **Interrupt Source Selection/Clearing Control** | |
| **DTCE** | **DISEL** | **DTC** | **CPU** |
| 0 | * | X | Δ |
| 1 | 0 | Δ | X |
| | 1 | ○ | Δ |

**Legend**

Δ : The relevant interrupt is used. Interrupt source clearing is performed.
(The CPU should clear the source flag in the interrupt handling routine.)

○ : The relevant interrupt is used. The interrupt source is not cleared.

X : The relevant bit cannot be used.

* : Don't care

**(4) Notes on Use:** SCI and A/D converter interrupt sources are cleared when the DTC reads or writes to the prescribed register, and are not dependent upon the DISEL bit.

# Section 6 Bus Controller

## 6.1 Overview

The H8S/2345 Series has a built-in bus controller (BSC) that manages the external address space divided into eight areas. The bus specifications, such as bus width and number of access states, can be set independently for each area, enabling multiple memories to be connected easily.

The bus controller also has a bus arbitration function, and controls the operation of the internal bus masters: the CPU and data transfer controller (DTC).

### 6.1.1 Features

The features of the bus controller are listed below.

- Manages external address space in area units
  - In advanced mode, manages the external space as 8 areas of 2-Mbytes
  - In normal mode*, manages the external space as a single area
  - Bus specifications can be set independently for each area
- Basic bus interface
  - Chip select ($\overline{CS0}$ to $\overline{CS3}$) can be output for areas 0 to 3
  - 8-bit access or 16-bit access can be selected for each area
  - 2-state access or 3-state access can be selected for each area
  - Program wait states can be inserted for each area
- Burst ROM interface
  - Burst ROM interface can be set for area 0
  - Choice of 1- or 2-state burst access
- Idle cycle insertion
  - An idle cycle can be inserted in case of an external read cycle between different areas
  - An idle cycle can be inserted in case of an external write cycle immediately after an external read cycle
- Bus arbitration function
  - Includes a bus arbiter that arbitrates bus mastership among the CPU and DTC
- Other features
  - External bus release function

Note: * ZTAT, mask ROM, and ROMless versions only.

### 6.1.2 Block Diagram

Figure 6.1 shows a block diagram of the bus controller.

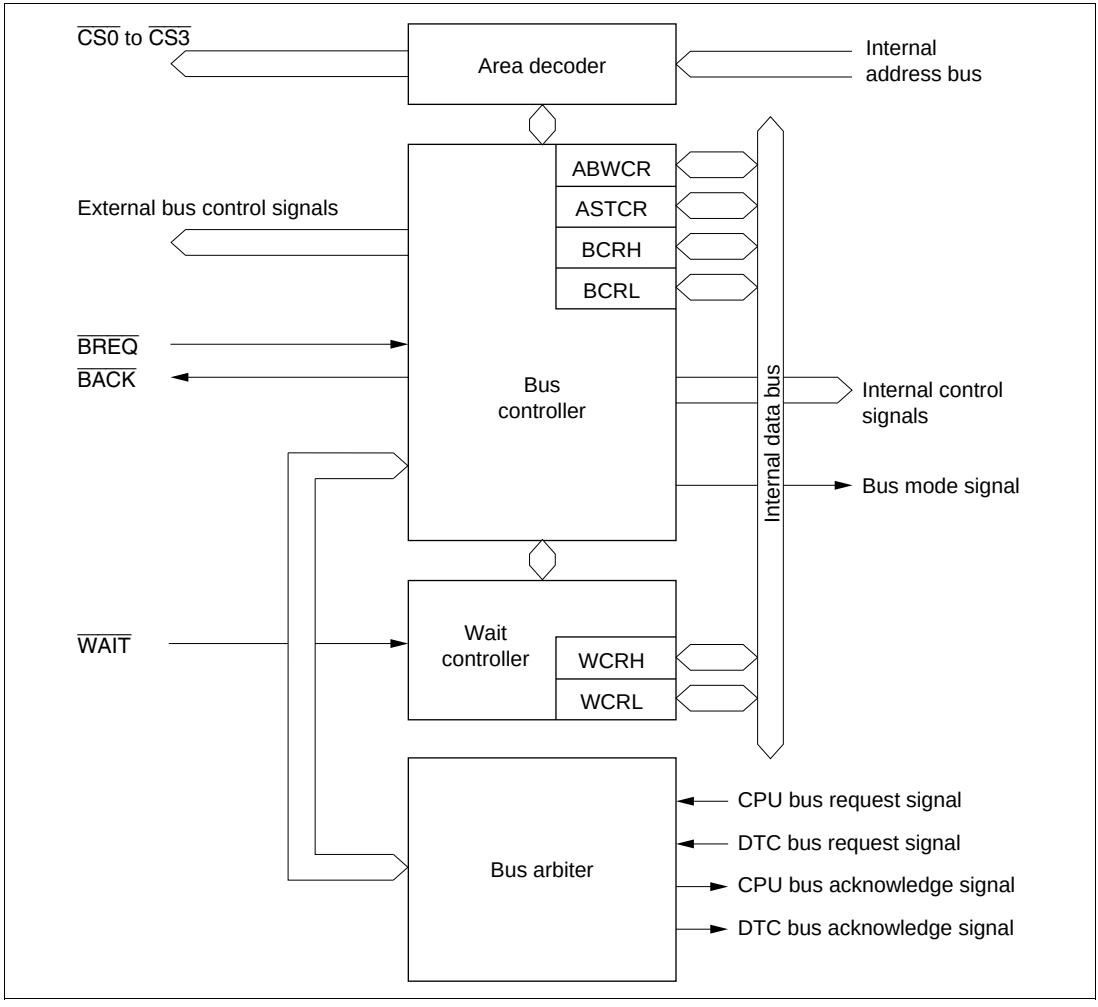



**Figure 6.1 Block Diagram of Bus Controller**

### 6.1.3 Pin Configuration

Table 6.1 summarizes the pins of the bus controller.

**Table 6.1 Bus Controller Pins**

| Name | Symbol | I/O | Function |
|---|---|---|---|
| Address strobe | $\overline{AS}$ | Output | Strobe signal indicating that address output on address bus is enabled. |
| Read | $\overline{RD}$ | Output | Strobe signal indicating that external space is being read. |
| High write | $\overline{HWR}$ | Output | Strobe signal indicating that external space is to be written, and upper half ($D_{15}$ to $D_8$) of data bus is enabled. |
| Low write | $\overline{LWR}$ | Output | Strobe signal indicating that external space is to be written, and lower half ($D_7$ to $D_0$) of data bus is enabled. |
| Chip select 0 to 3 | $\overline{CS0}$ to $\overline{CS3}$ | Output | Strobe signal indicating that areas 0 to 3 are selected. |
| Wait | $\overline{WAIT}$ | Input | Wait request signal when accessing external 3-state access space. |
| Bus request | $\overline{BREQ}$ | Input | Request signal that releases bus to external device. |
| Bus request acknowledge | $\overline{BACK}$ | Output | Acknowledge signal indicating that bus has been released. |

### 6.1.4 Register Configuration

Table 6.2 summarizes the registers of the bus controller.

**Table 6.2 Bus Controller Registers**

| | | | Initial Value | | |
|---|---|---|---|---|---|
| Name | Abbreviation | R/W | Power-On Reset | Manual Reset | Address*[1] |
| Bus width control register | ABWCR | R/W | H'FF/H'00*[2] | Retained | H'FED0 |
| Access state control register | ASTCR | R/W | H'FF | Retained | H'FED1 |
| Wait control register H | WCRH | R/W | H'FF | Retained | H'FED2 |
| Wait control register L | WCRL | R/W | H'FF | Retained | H'FED3 |
| Bus control register H | BCRH | R/W | H'D0 | Retained | H'FED4 |
| Bus control register L | BCRL | R/W | H'3C | Retained | H'FED5 |

Notes: 1. Lower 16 bits of the address.
2. Determined by the MCU operating mode.

## 6.2 Register Descriptions

### 6.2.1 Bus Width Control Register (ABWCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | ABW7 | ABW6 | ABW5 | ABW4 | ABW3 | ABW2 | ABW1 | ABW0 |
| Modes 1 to 3*, 5 to 7 | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| RW | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Mode 4 | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| RW | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

ABWCR is an 8-bit readable/writable register that designates each area for either 8-bit access or 16-bit access.

ABWCR sets the data bus width for the external memory space. The bus width for on-chip memory and internal I/O registers is fixed regardless of the settings in ABWCR.

In normal mode*, the settings of bits ABW7 to ABW1 have no effect on operation.

After a power-on reset and in hardware standby mode, ABWCR is initialized to H'FF in modes 1, 2, 3*, and 5, 6, 7, and to H'00 in mode 4. It is not initialized by a manual reset or in software standby mode.

**Bits 7 to 0—Area 7 to 0 Bus Width Control (ABW7 to ABW0):** These bits select whether the corresponding area is to be designated for 8-bit access or 16-bit access. In normal mode*, only part of area 0 is enabled, and the ABW0 bit selects whether external space is to be designated for 8-bit access or 16-bit access.

Note: * ZTAT, mask ROM, and ROMless versions only.

| Bit n | |
|---|---|
| **ABWn** | **Description** |
| 0 | Area n is designated for 16-bit access |
| 1 | Area n is designated for 8-bit access |

(n = 7 to 0)

### 6.2.2 Access State Control Register (ASTCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | AST7 | AST6 | AST5 | AST4 | AST3 | AST2 | AST1 | AST0 |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

ASTCR is an 8-bit readable/writable register that designates each area as either a 2-state access space or a 3-state access space.

ASTCR sets the number of access states for the external memory space. The number of access states for on-chip memory and internal I/O registers is fixed regardless of the settings in ASTCR.

In normal mode*, the settings of bits AST7 to AST1 have no effect on operation.

ASTCR is initialized to H'FF by a power-on reset and in hardware standby mode. It is not initialized by a manual reset or in software standby mode.

**Bits 7 to 0—Area 7 to 0 Access State Control (AST7 to AST0):** These bits select whether the corresponding area is to be designated as a 2-state access space or a 3-state access space. In normal mode*, only part of area 0 is enabled, and the AST0 bit selects whether external space is to be designated for 2-state access or 3-state access.

Wait state insertion is enabled or disabled at the same time.

Note: * ZTAT, mask ROM, and ROMless versions only.

| Bit n<br>ASTn | Description | |
|---|---|---|
| 0 | Area n is designated for 2-state access<br>Wait state insertion in area n external space is disabled | |
| 1 | Area n is designated for 3-state access<br>Wait state insertion in area n external space is enabled | (Initial value) |

(n = 7 to 0)

### 6.2.3 Wait Control Registers H and L (WCRH, WCRL)

WCRH and WCRL are 8-bit readable/writable registers that select the number of program wait states for each area.

In normal mode*, only part of area is 0 is enabled, and bits W01 and W00 select the number of program wait states for the external space . The settings of bits W71, W70 to W11, and W10 have no effect on operation.

Program waits are not inserted in the case of on-chip memory or internal I/O registers.

WCRH and WCRL are initialized to H'FF by a power-on reset and in hardware standby mode. They are not initialized by a manual reset or in software standby mode.

Note: * ZTAT, mask ROM, and ROMless versions only.

**(1) WCRH**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | W71 | W70 | W61 | W60 | W51 | W50 | W41 | W40 |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Bits 7 and 6—Area 7 Wait Control 1 and 0 (W71, W70):** These bits select the number of program wait states when area 7 in external space is accessed while the AST7 bit in ASTCR is set to 1.

| Bit 7 | Bit 6 | |
|---|---|---|
| **W71** | **W70** | **Description** |
| 0 | 0 | Program wait not inserted when external space area 7 is accessed |
| | 1 | 1 program wait state inserted when external space area 7 is accessed |
| 1 | 0 | 2 program wait states inserted when external space area 7 is accessed |
| | 1 | 3 program wait states inserted when external space area 7 is accessed (Initial value) |

**Bits 5 and 4—Area 6 Wait Control 1 and 0 (W61, W60):** These bits select the number of program wait states when area 6 in external space is accessed while the AST6 bit in ASTCR is set to 1.

| Bit 5 | Bit 4 | |
|---|---|---|
| **W61** | **W60** | **Description** |
| 0 | 0 | Program wait not inserted when external space area 6 is accessed |
| | 1 | 1 program wait state inserted when external space area 6 is accessed |
| 1 | 0 | 2 program wait states inserted when external space area 6 is accessed |
| | 1 | 3 program wait states inserted when external space area 6 is accessed (Initial value) |

**Bits 3 and 2—Area 5 Wait Control 1 and 0 (W51, W50):** These bits select the number of program wait states when area 5 in external space is accessed while the AST5 bit in ASTCR is set to 1.

| Bit 3 | Bit 2 | |
|---|---|---|
| **W51** | **W50** | **Description** |
| 0 | 0 | Program wait not inserted when external space area 5 is accessed |
| | 1 | 1 program wait state inserted when external space area 5 is accessed |
| 1 | 0 | 2 program wait states inserted when external space area 5 is accessed |
| | 1 | 3 program wait states inserted when external space area 5 is accessed (Initial value) |

**Bits 1 and 0—Area 4 Wait Control 1 and 0 (W41, W40):** These bits select the number of program wait states when area 4 in external space is accessed while the AST4 bit in ASTCR is set to 1.

| Bit 1 | Bit 0 | |
|---|---|---|
| **W41** | **W40** | **Description** |
| 0 | 0 | Program wait not inserted when external space area 4 is accessed |
| | 1 | 1 program wait state inserted when external space area 4 is accessed |
| 1 | 0 | 2 program wait states inserted when external space area 4 is accessed |
| | 1 | 3 program wait states inserted when external space area 4 is accessed (Initial value) |

**(2) WCRL**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | W31 | W30 | W21 | W20 | W11 | W10 | W01 | W00 |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Bits 7 and 6—Area 3 Wait Control 1 and 0 (W31, W30):** These bits select the number of program wait states when area 3 in external space is accessed while the AST3 bit in ASTCR is set to 1.

| Bit 7 | Bit 6 | |
|---|---|---|
| **W31** | **W30** | **Description** |
| 0 | 0 | Program wait not inserted when external space area 3 is accessed |
| | 1 | 1 program wait state inserted when external space area 3 is accessed |
| 1 | 0 | 2 program wait states inserted when external space area 3 is accessed |
| | 1 | 3 program wait states inserted when external space area 3 is accessed (Initial value) |

**Bits 5 and 4—Area 2 Wait Control 1 and 0 (W21, W20):** These bits select the number of program wait states when area 2 in external space is accessed while the AST2 bit in ASTCR is set to 1.

| Bit 5 | Bit 4 | |
|---|---|---|
| **W21** | **W20** | **Description** |
| 0 | 0 | Program wait not inserted when external space area 2 is accessed |
| | 1 | 1 program wait state inserted when external space area 2 is accessed |
| 1 | 0 | 2 program wait states inserted when external space area 2 is accessed |
| | 1 | 3 program wait states inserted when external space area 2 is accessed (Initial value) |

**Bits 3 and 2—Area 1 Wait Control 1 and 0 (W11, W10):** These bits select the number of program wait states when area 1 in external space is accessed while the AST1 bit in ASTCR is set to 1.

| Bit 3 | Bit 2 | |
|---|---|---|
| **W11** | **W10** | **Description** |
| 0 | 0 | Program wait not inserted when external space area 1 is accessed |
| | 1 | 1 program wait state inserted when external space area 1 is accessed |
| 1 | 0 | 2 program wait states inserted when external space area 1 is accessed |
| | 1 | 3 program wait states inserted when external space area 1 is accessed (Initial value) |

**Bits 1 and 0—Area 0 Wait Control 1 and 0 (W01, W00):** These bits select the number of program wait states when area 0 in external space is accessed while the AST0 bit in ASTCR is set to 1.

| Bit 1 | Bit 0 | |
|---|---|---|
| **W01** | **W00** | **Description** |
| 0 | 0 | Program wait not inserted when external space area 0 is accessed |
| | 1 | 1 program wait state inserted when external space area 0 is accessed |
| 1 | 0 | 2 program wait states inserted when external space area 0 is accessed |
| | 1 | 3 program wait states inserted when external space area 0 is accessed (Initial value) |

### 6.2.4 Bus Control Register H (BCRH)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | ICIS1 | ICIS0 | BRSTRM | BRSTS1 | BRSTS0 | — | — | — |
| Initial value | : | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

BCRH is an 8-bit readable/writable register that selects enabling or disabling of idle cycle insertion, and the memory interface for areas 2 to 5 and area 0.

BCRH is initialized to H'D0 by a power-on reset and in hardware standby mode. It is not initialized by a manual reset or in software standby mode.

**Bit 7—Idle Cycle Insert 1 (ICIS1):** Selects whether or not one idle cycle state is to be inserted between bus cycles when successive external read cycles are performed in different areas.

| Bit 7 | |
|---|---|
| **ICIS1** | **Description** |
| 0 | Idle cycle not inserted in case of successive external read cycles in different areas |
| 1 | Idle cycle inserted in case of successive external read cycles in different areas (Initial value) |

**Bit 6—Idle Cycle Insert 0 (ICIS0):** Selects whether or not one idle cycle state is to be inserted between bus cycles when successive external read and external write cycles are performed .

| Bit 6 | |
|---|---|
| **ICIS0** | **Description** |
| 0 | Idle cycle not inserted in case of successive external read and external write cycles |
| 1 | Idle cycle inserted in case of successive external read and external write cycles (Initial value) |

**Bit 5—Burst ROM Enable (BRSTRM):** Selects whether area 0 is used as a burst ROM interface. In normal mode*, the selection can be made from the entire external space .

Burst ROM interface and PSRAM burst operation cannot be set at the same time.

Note: * ZTAT, mask ROM, and ROMless versions only.

| Bit 5 | | |
|---|---|---|
| **BRSTRM** | **Description** | |
| 0 | Area 0 is basic bus interface | (Initial value) |
| 1 | Area 0 is burst ROM interface | |

**Bit 4—Burst Cycle Select 1 (BRSTS1):** Selects the number of burst cycles for the burst ROM interface.

| Bit 4 | | |
|---|---|---|
| **BRSTS1** | **Description** | |
| 0 | Burst cycle comprises 1 state | |
| 1 | Burst cycle comprises 2 states | (Initial value) |

**Bit 3—Burst Cycle Select 0 (BRSTS0):** Selects the number of words that can be accessed in a burst ROM interface burst access.

| Bit 3 | |
|---|---|
| **BRSTS0** | **Description** |
| 0 | Max. 4 words in burst access (Initial value) |
| 1 | Max. 8 words in burst access |

**Bits 2 to 0—Reserved:** Only 0 should be written to these bits.

### 6.2.5 Bus Control Register L (BCRL)

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BRLE | — | EAE | — | — | — | — | WAITE |
| Initial value : | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

BCRL is an 8-bit readable/writable register that performs selection of the external bus-released state protocol, and enabling or disabling of $\overline{\text{WAIT}}$ pin input.

BCRL is initialized to H'3C by a power-on reset and in hardware standby mode. It is not initialized by a manual reset or in software standby mode.

**Bit 7—Bus Release Enable (BRLE):** Enables or disables external bus release.

| Bit 7 | |
|---|---|
| **BRLE** | **Description** |
| 0 | External bus release is disabled. $\overline{\text{BREQ}}$ and $\overline{\text{BACK}}$ can be used as I/O ports. (Initial value) |
| 1 | External bus release is enabled. |

**Bit 6—Reserved:** Only 0 should be written to this bit.

**Bit 5—External Address Enable (EAE):** Selects whether addresses H'010000 to H'01FFFF are to be internal addresses or external addresses.

This setting is invalid in normal mode*.

Note: * ZTAT, mask ROM, and ROMless versions only.

**Bit 5**

| EAE | Description |
|---|---|
| 0 | Addresses H'010000 to H'01FFFF are in on-chip ROM (in the H8S/2345)<br>Addresses H'010000 to H'017FFF are in on-chip ROM and addresses H'018000 to H'01FFFF are a reserved area (in the H8S/2344)<br>Addresses H'010000 to H'01FFFF are a reserved area (in the H8S/2343 and H8S/2341) |
| 1 | Addresses H'010000 to H'01FFFF are external addresses (external expansion mode) or a reserved area* (single-chip mode) (Initial value) |

Note: * Reserved areas should not be accessed.

**Bits 4 to 2—Reserved:** Only 1 should be written to these bits.

**Bit 1—Reserved:** Only 0 should be written to this bit.

**Bit 0—WAIT Pin Enable (WAITE):** Selects enabling or disabling of wait input by the $\overline{\text{WAIT}}$ pin.

**Bit 0**

| WAITE | Description |
|---|---|
| 0 | Wait input by $\overline{\text{WAIT}}$ pin disabled. $\overline{\text{WAIT}}$ pin can be used as I/O port. (Initial value) |
| 1 | Wait input by $\overline{\text{WAIT}}$ pin enabled |

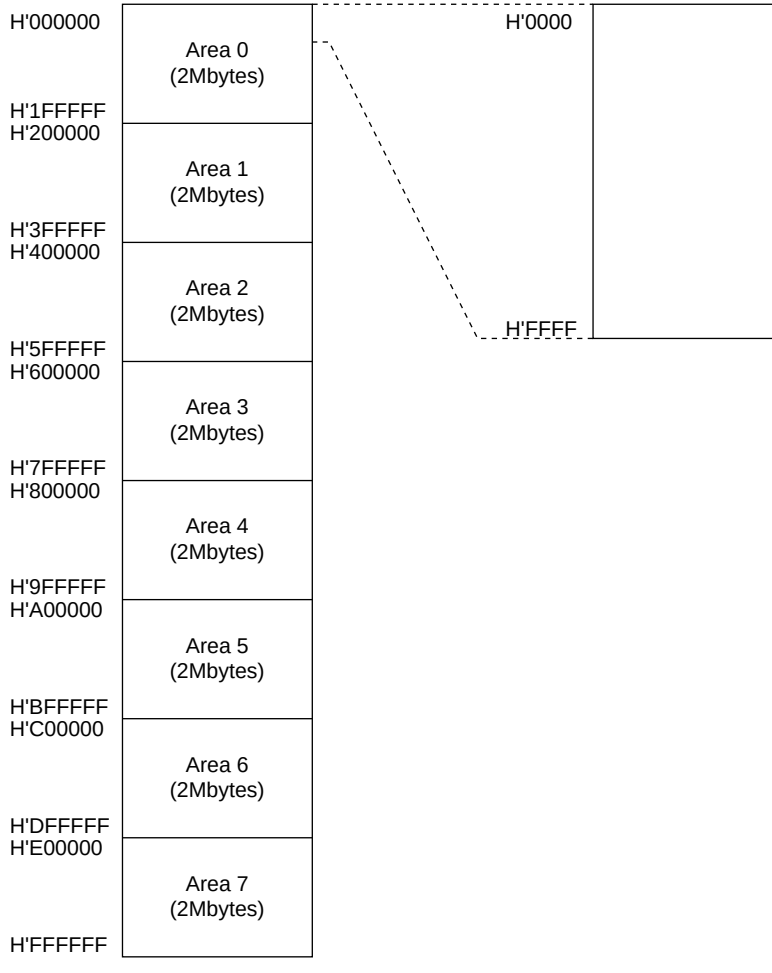
## 6.3 Overview of Bus Control

### 6.3.1 Area Partitioning

In advanced mode, the bus controller partitions the 16 Mbytes address space into eight areas, 0 to 7, in 2-Mbyte units, and performs bus control for external space in area units. In normal mode*, it controls a 64-kbyte address space comprising part of area 0. Figure 6.2 shows an outline of the memory map.

Chip select signals ($\overline{\text{CS0}}$ to $\overline{\text{CS3}}$) can be output for areas 0 to 3.

Note: * ZTAT, mask ROM, and ROMless versions only.




Note: * ZTAT, mask ROM, and ROMless versions only.

**Figure 6.2 Overview of Area Partitioning**

### 6.3.2 Bus Specifications

The external space bus specifications consist of three elements: bus width, number of access states, and number of program wait states.

The bus width and number of access states for on-chip memory and internal I/O registers are fixed, and are not affected by the bus controller.

**(1) Bus Width:** A bus width of 8 or 16 bits can be selected with ADWCR. An area for which an 8-bit bus is selected functions as an 8-bit access space, and an area for which a 16-bit bus is selected functions as a16-bit access space.

If all areas are designated for 8-bit access, 8-bit bus mode is set; if any area is designated for 16-bit access, 16-bit bus mode is set. When the burst ROM interface is designated, 16-bit bus mode is always set.

**(2) Number of Access States:** Two or three access states can be selected with ASTCR. An area for which 2-state access is selected functions as a 2-state access space, and an area for which 3-state access is selected functions as a 3-state access space.

With the burst ROM interface, the number of access states may be determined without regard to ASTCR.

When 2-state access space is designated, wait insertion is disabled.

**(3) Number of Program Wait States:** When 3-state access space is designated by ASTCR, the number of program wait states to be inserted automatically is selected with WCRH and WCRL. From 0 to 3 program wait states can be selected.

Table 6.3 shows the bus specifications for each basic bus interface area.

**Table 6.3 Bus Specifications for Each Area (Basic Bus Interface)**

| ABWCR | ASTCR | WCRH, WCRL | | Bus Specifications (Basic Bus Interface) | | |
|---|---|---|---|---|---|---|
| ABWn | ASTn | Wn1 | Wn0 | Bus Width | Access States | Program Wait States |
| 0 | 0 | — | — | 16 | 2 | 0 |
| | 1 | 0 | 0 | | 3 | 0 |
| | | | 1 | | | 1 |
| | | 1 | 0 | | | 2 |
| | | | 1 | | | 3 |
| 1 | 0 | — | — | 8 | 2 | 0 |
| | 1 | 0 | 0 | | 3 | 0 |
| | | | 1 | | | 1 |
| | | 1 | 0 | | | 2 |
| | | | 1 | | | 3 |

### 6.3.3 Memory Interfaces

The H8S/2345 Series memory interfaces comprise a basic bus interface that allows direct connection of ROM, SRAM, and so on, and a burst ROM interface (for area 0 only) that allows direct connection of burst ROM.

An area for which the basic bus interface is designated functions as normal space, and an area for which the burst ROM interface is designated functions as burst ROM space.

### 6.3.4 Advanced Mode

The initial state of each area is basic bus interface, 3-state access space. The initial bus width is selected according to the operating mode. The bus specifications described here cover basic items only, and the sections on each memory interface (6.4 and 6.5) should be referred to for further details.

**Area 0:** Area 0 includes on-chip ROM, and in ROM-disabled expansion mode, all of area 0 is external space. In ROM-enabled expansion mode, the space excluding on-chip ROM is external space.

When area 0 external space is accessed, the $\overline{CS0}$ signal can be output.

Either basic bus interface or burst ROM interface can be selected for area 0.

**Areas 1 to 6:** In external expansion mode, all of areas 1 to 6 is external space.

When area 1 to 3 external space is accessed, the $\overline{CS1}$ to $\overline{CS3}$ pin signals respectively can be output.

Only the basic bus interface can be used for areas 1 to 6.

**Area 7:** Area 7 includes the on-chip RAM and internal I/O registers. In external expansion mode, the space excluding the on-chip RAM and internal I/O registers is external space. The on-chip RAM is enabled when the RAME bit in the system control register (SYSCR) is set to 1; when the RAME bit is cleared to 0, the on-chip RAM is disabled and the corresponding space becomes external space.

Only the basic bus interface can be used for the area 7 memory interface.

### 6.3.5 Areas in Normal Mode (ZTAT, Mask ROM, and ROMless versions Only)

In normal mode, a 64-kbyte address space comprising part of area 0 is controlled. Area partitioning is not performed in normal mode. In ROM-disabled expansion mode, the space excluding the on-chip RAM and internal I/O registers is external space. In ROM-enabled expansion mode the space excluding the on-chip ROM, on-chip RAM, and internal I/O registers is external space. The on-chip RAM is enabled when the RAME bit in the system control register (SYSCR) is set to 1; when the RAME bit is cleared to 0, the on-chip RAM is disabled and the corresponding space becomes external space .

When external space is accessed, the $\overline{CS0}$ signal can be output.

The basic bus interface or burst ROM interface can be selected.

### 6.3.6 Chip Select Signals

The H8S/2345 Series can output chip select signals ($\overline{CS0}$ to $\overline{CS3}$) to areas 0 to 3, the signal being driven low when the corresponding external space area is accessed. In normal mode*, only the $\overline{CS0}$ signal can be output.

Figure 6.3 shows an example of $\overline{CSn}$ (n = 0 to 3) output timing.

Enabling or disabling of the $\overline{CSn}$ signal is performed by setting the data direction register (DDR) for the port corresponding to the particular $\overline{CSn}$ pin.

In ROM-disabled expansion mode, the $\overline{CS0}$ pin is placed in the output state after a power-on reset. Pins $\overline{CS1}$ to $\overline{CS3}$ are placed in the input state after a power-on reset, and so the corresponding DDR should be set to 1 when outputting signals $\overline{CS1}$ to $\overline{CS3}$.

In ROM-enabled expansion mode, pins $\overline{CS0}$ to $\overline{CS3}$ are all placed in the input state after a power-on reset, and so the corresponding DDR should be set to 1 when outputting signals $\overline{CS0}$ to $\overline{CS3}$.

For details, see section 8, I/O Ports.

Note: * ZTAT, mask ROM, and ROMless versions only.

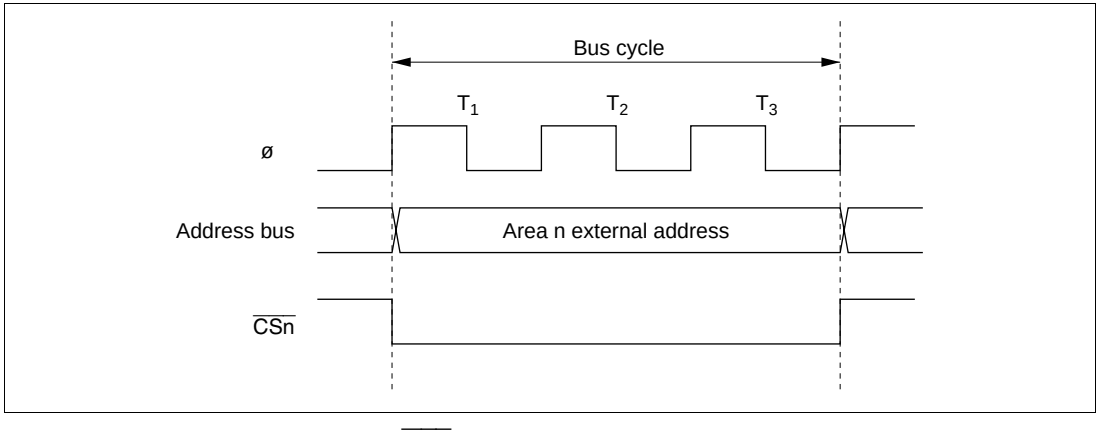



**Figure 6.3 $\overline{CSn}$ Signal Output Timing (n = 0 to 3)**

## 6.4 Basic Bus Interface

### 6.4.1 Overview

The basic bus interface enables direct connection of ROM, SRAM, and so on.

The bus specifications can be selected with ABWCR, ASTCR, WCRH, and WCRL (see table 6.3).

### 6.4.2 Data Size and Data Alignment

Data sizes for the CPU and other internal bus masters are byte, word, and longword. The bus controller has a data alignment function, and when accessing external space, controls whether the upper data bus ($D_{15}$ to $D_8$) or lower data bus ($D_7$ to $D_0$) is used according to the bus specifications for the area being accessed (8-bit access space or 16-bit access space) and the data size.

**8-Bit Access Space:** Figure 6.4 illustrates data alignment control for the 8-bit access space. With the 8-bit access space, the upper data bus ($D_{15}$ to $D_8$) is always used for accesses. The amount of data that can be accessed at one time is one byte: a word transfer instruction is performed as two byte accesses, and a longword transfer instruction, as four byte accesses.

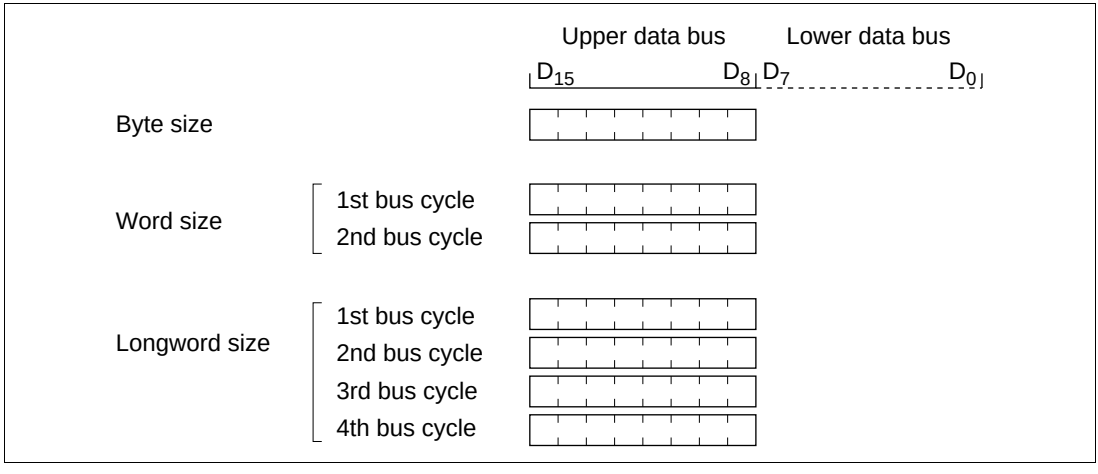



**Figure 6.4 Access Sizes and Data Alignment Control (8-Bit Access Space)**

**16-Bit Access Space:** Figure 6.5 illustrates data alignment control for the 16-bit access space. With the 16-bit access space, the upper data bus ($D_{15}$ to $D_8$) and lower data bus ($D_7$ to $D_0$) are used for accesses. The amount of data that can be accessed at one time is one byte or one word, and a longword transfer instruction is executed as two word transfer instructions.

In byte access, whether the upper or lower data bus is used is determined by whether the address is even or odd. The upper data bus is used for an even address, and the lower data bus for an odd address.

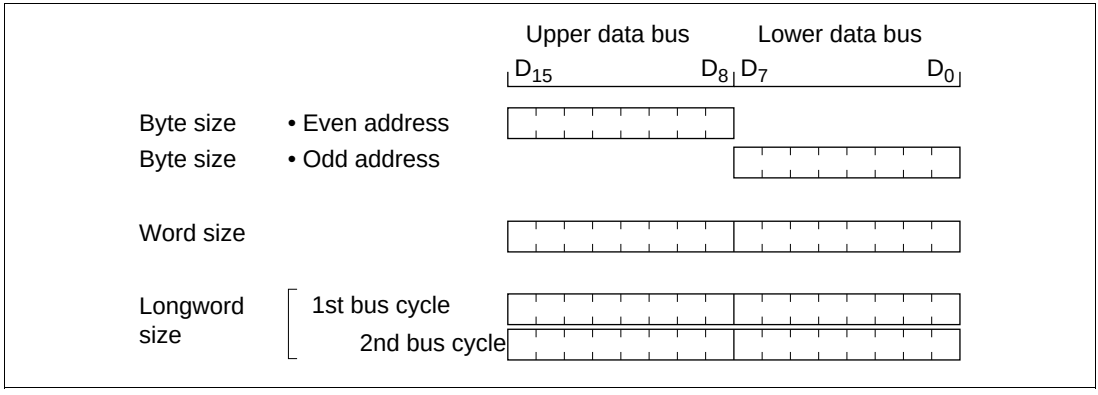



**Figure 6.5 Access Sizes and Data Alignment Control (16-Bit Access Space)**

### 6.4.3 Valid Strobes

Table 6.4 shows the data buses used and valid strobes for the access spaces.

In a read, the $\overline{RD}$ signal is valid without discrimination between the upper and lower halves of the data bus.

In a write, the $\overline{HWR}$ signal is valid for the upper half of the data bus, and the $\overline{LWR}$ signal for the lower half.

**Table 6.4 Data Buses Used and Valid Strobes**

| Area | Access Size | Read/ Write | Address | Valid Strobe | Upper Data Bus ($D_{15}$ to $D_8$) | Lower data bus ($D_7$ to $D_0$) |
|---|---|---|---|---|---|---|
| 8-bit access space | Byte | Read | — | $\overline{RD}$ | Valid | Invalid |
| | | Write | — | $\overline{HWR}$ | | Hi-Z |
| 16-bit access space | Byte | Read | Even | $\overline{RD}$ | Valid | Invalid |
| | | | Odd | | Invalid | Valid |
| | | Write | Even | $\overline{HWR}$ | Valid | Hi-Z |
| | | | Odd | $\overline{LWR}$ | Hi-Z | Valid |
| | Word | Read | — | $\overline{RD}$ | Valid | Valid |
| | | Write | — | $\overline{HWR}$, $\overline{LWR}$ | Valid | Valid |

Note: Hi-Z: High impedance.
Invalid: Input state; input value is ignored.

### 6.4.4 Basic Timing

**8-Bit 2-State Access Space:** Figure 6.6 shows the bus timing for an 8-bit 2-state access space. When an 8-bit access space is accessed , the upper half ($D_{15}$ to $D_8$) of the data bus is used.

The $\overline{\text{LWR}}$ pin is fixed high. Wait states cannot be inserted.

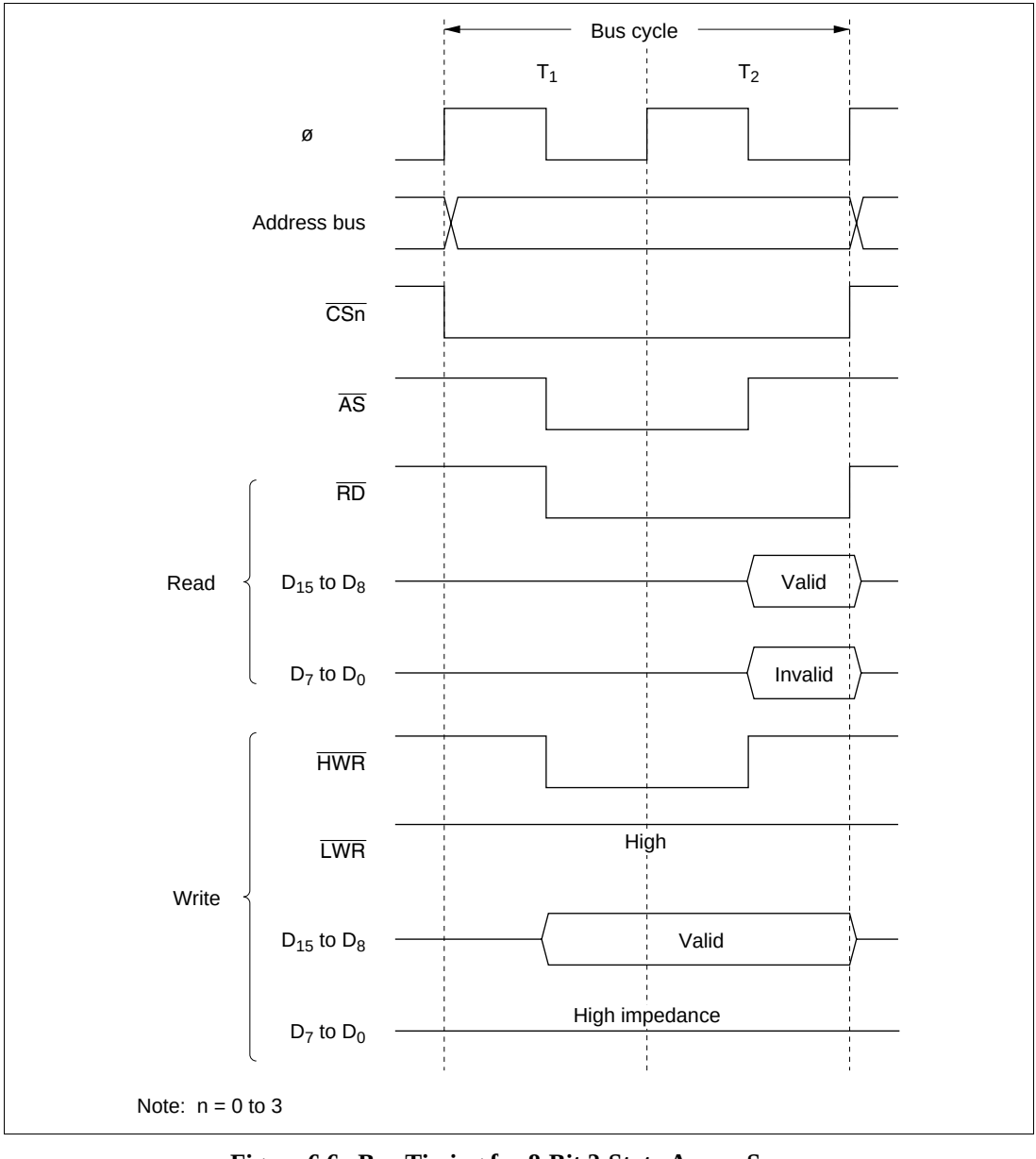



**Figure 6.6 Bus Timing for 8-Bit 2-State Access Space**

**8-Bit 3-State Access Space:** Figure 6.7 shows the bus timing for an 8-bit 3-state access space. When an 8-bit access space is accessed, the upper half ($D_{15}$ to $D_8$) of the data bus is used.

The $\overline{LWR}$ pin is fixed high. Wait states can be inserted.

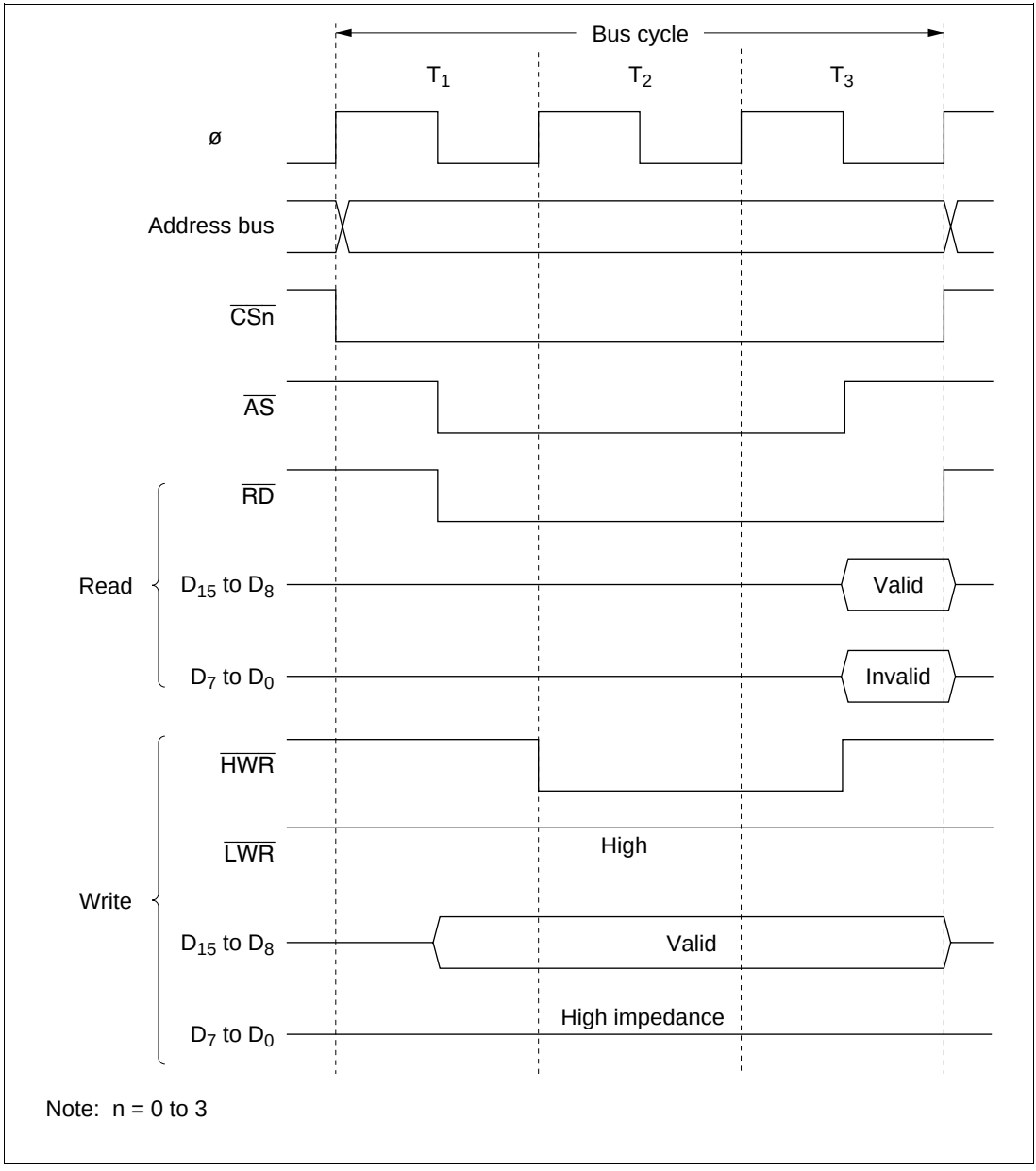



**Figure 6.7 Bus Timing for 8-Bit 3-State Access Space**

**16-Bit 2-State Access Space:** Figures 6.8 to 6.10 show bus timings for a 16-bit 2-state access space. When a 16-bit access space is accessed, the upper half ($D_{15}$ to $D_8$) of the data bus is used for the even address, and the lower half ($D_7$ to $D_0$) for the odd address.

Wait states cannot be inserted.

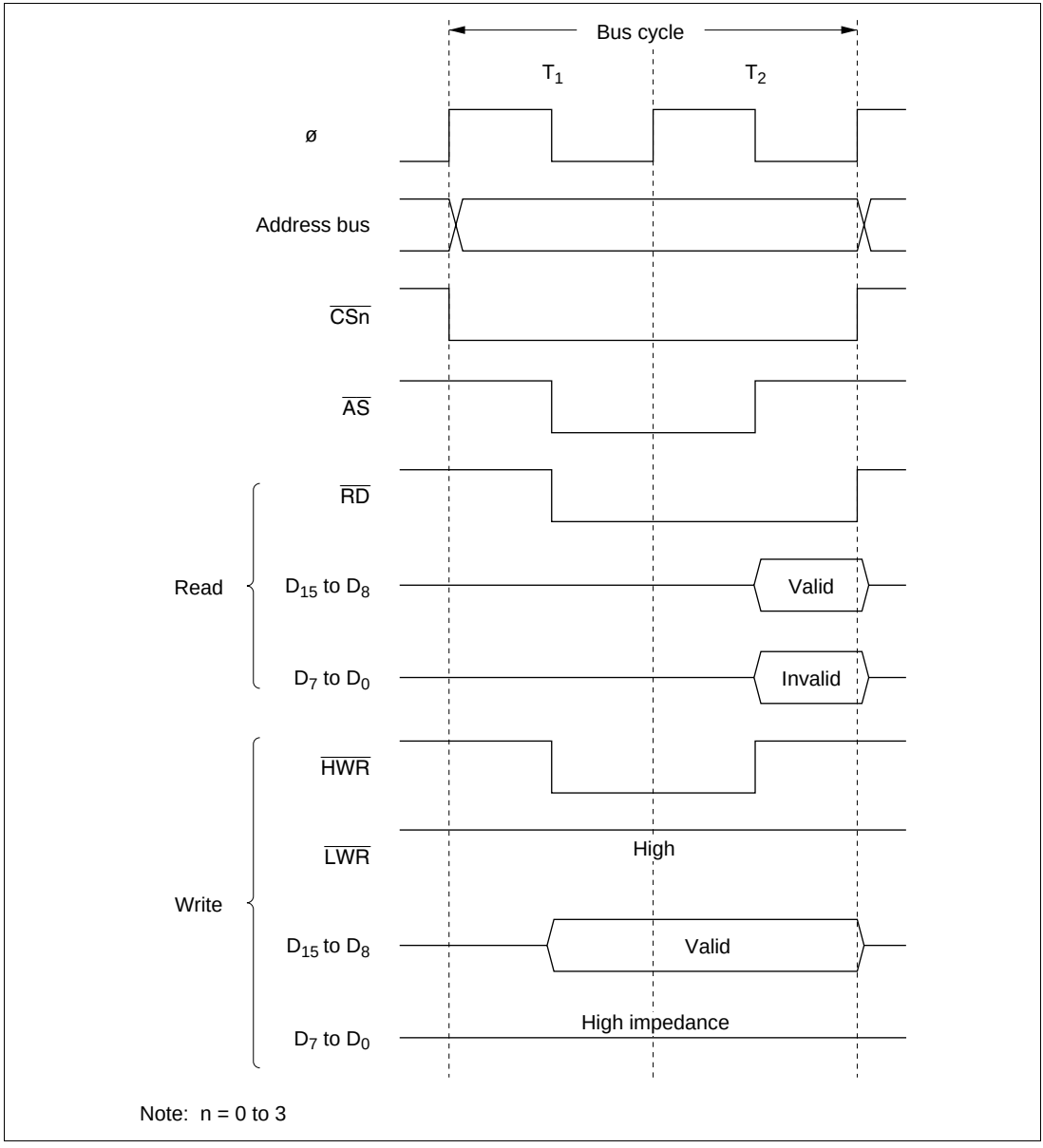



**Figure 6.8 Bus Timing for 16-Bit 2-State Access Space (1) (Even Address Byte Access)**

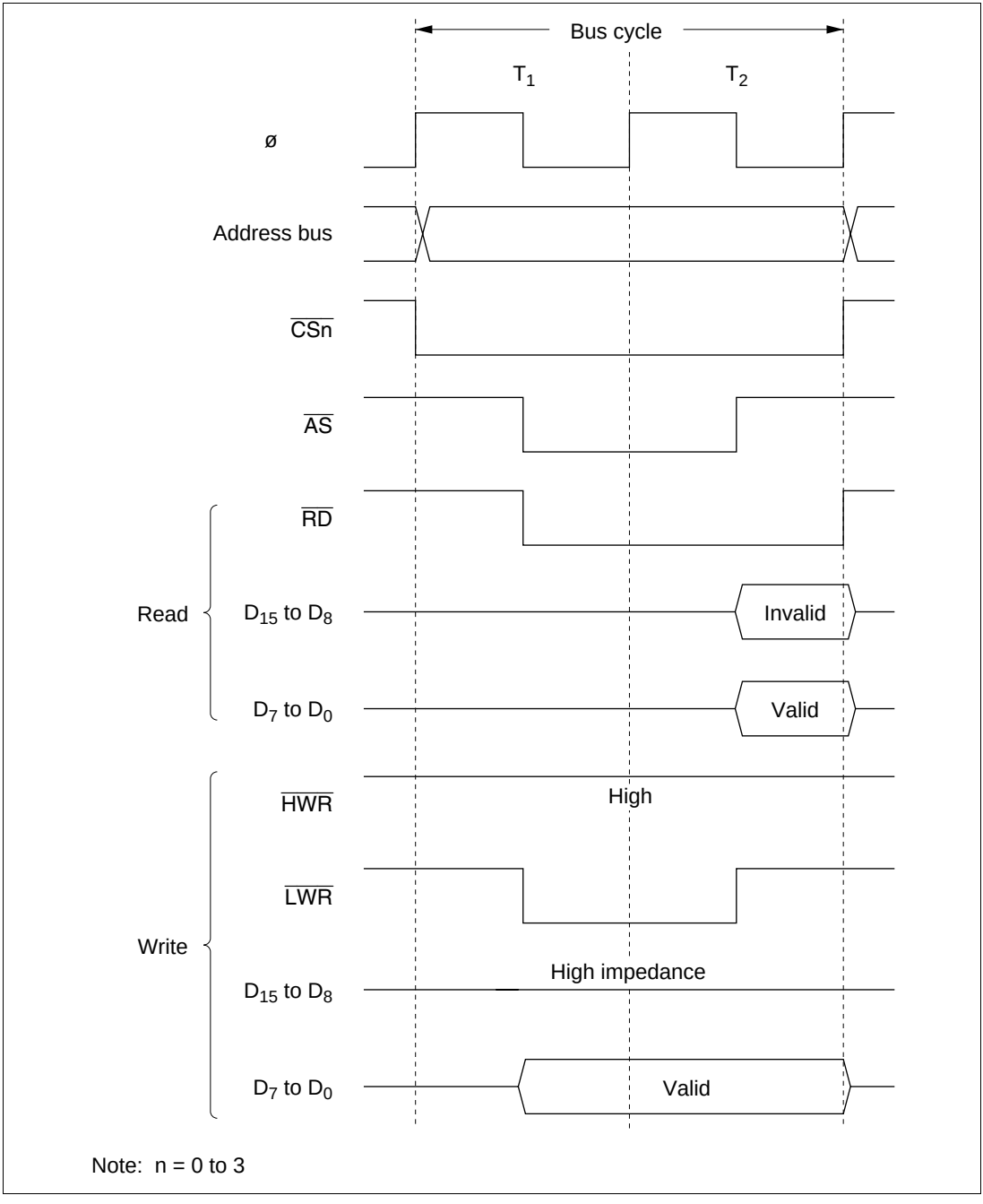



**Figure 6.9 Bus Timing for 16-Bit 2-State Access Space (2) (Odd Address Byte Access)**

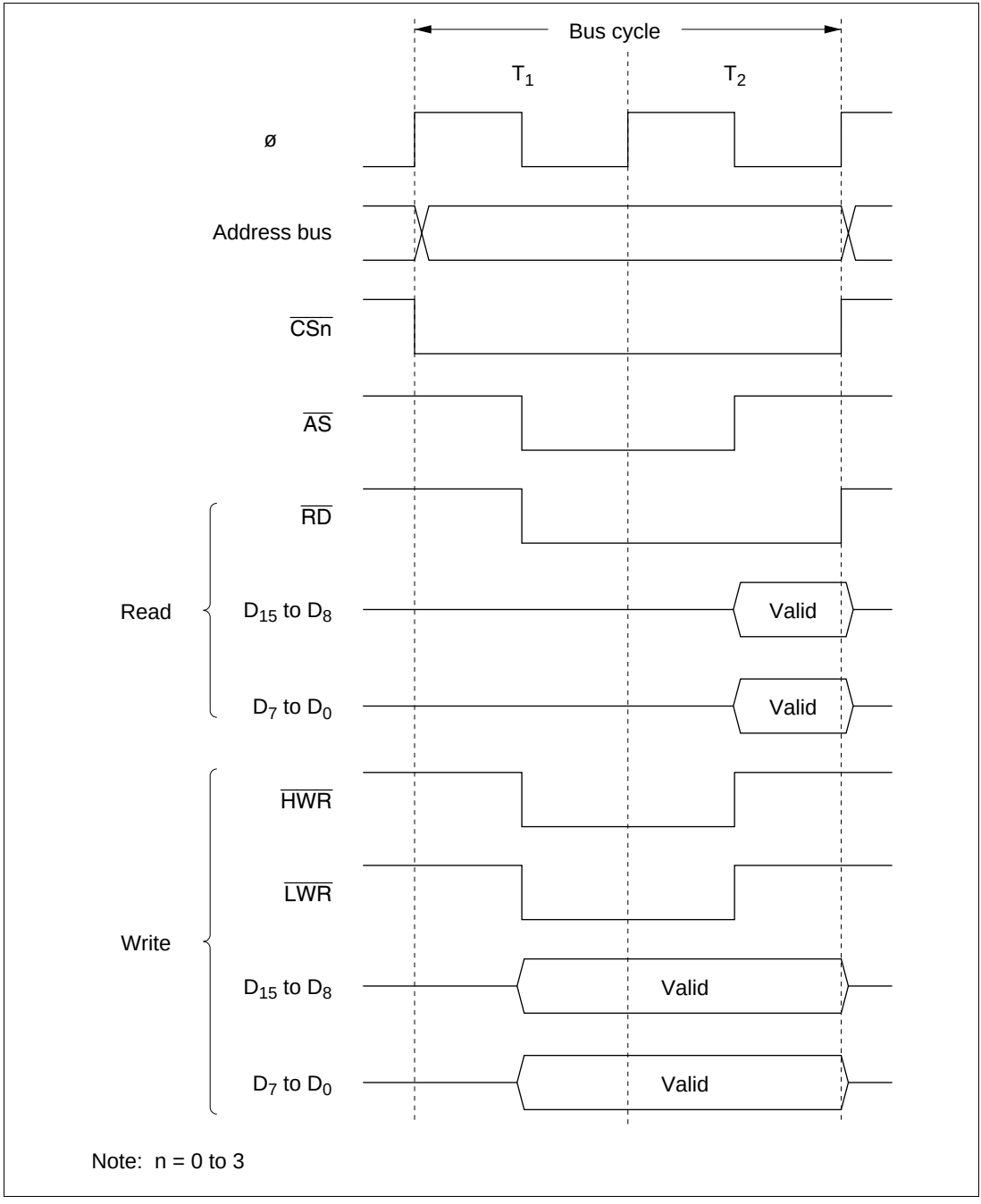



**Figure 6.10 Bus Timing for 16-Bit 2-State Access Space (3) (Word Access)**

**16-Bit 3-State Access Space:** Figures 6.11 to 6.13 show bus timings for a 16-bit 3-state access space. When a 16-bit access space is accessed , the upper half ($D_{15}$ to $D_8$) of the data bus is used for the even address, and the lower half ($D_7$ to $D_0$) for the odd address.

Wait states can be inserted.

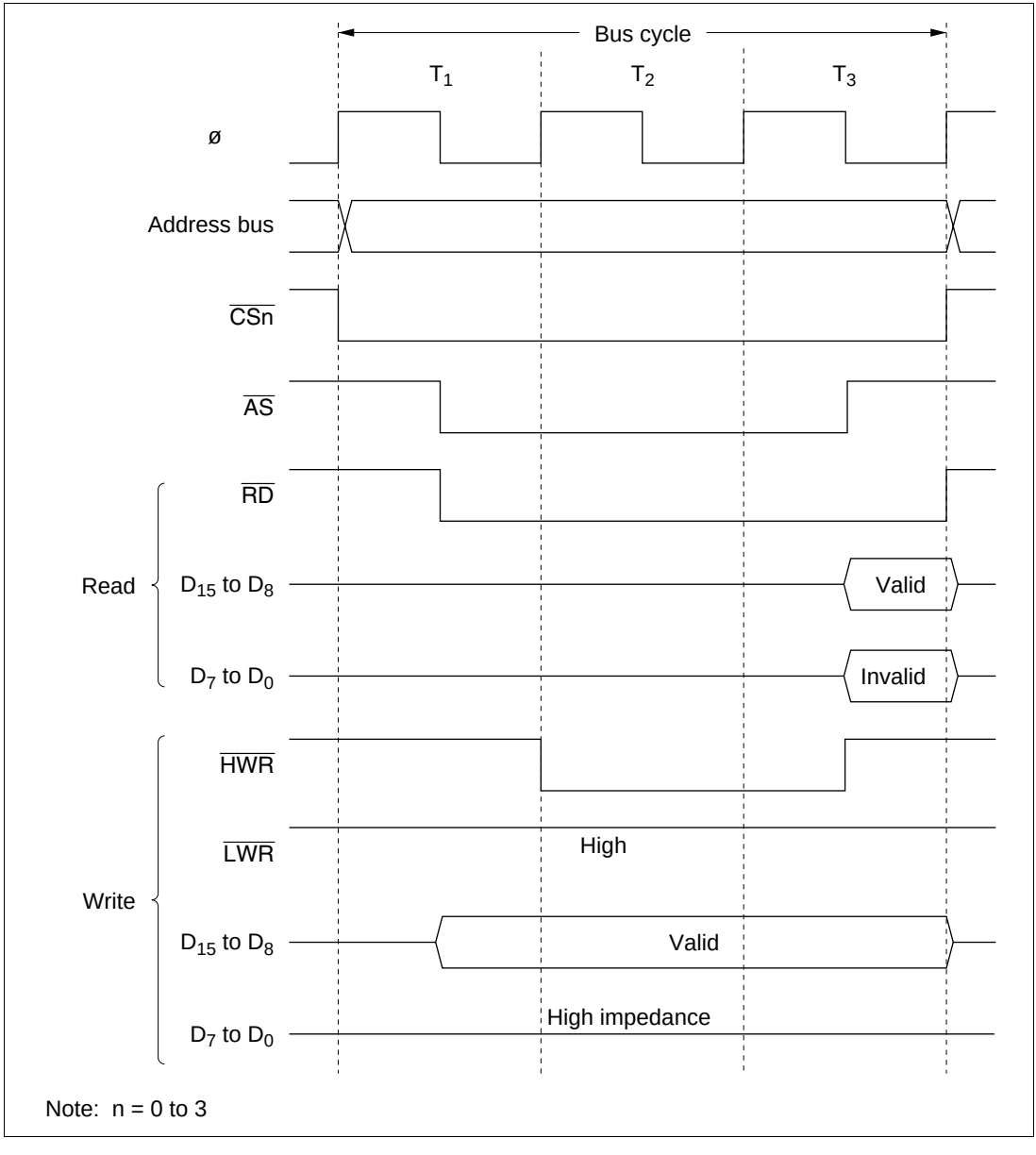



**Figure 6.11 Bus Timing for 16-Bit 3-State Access Space (1) (Even Address Byte Access)**

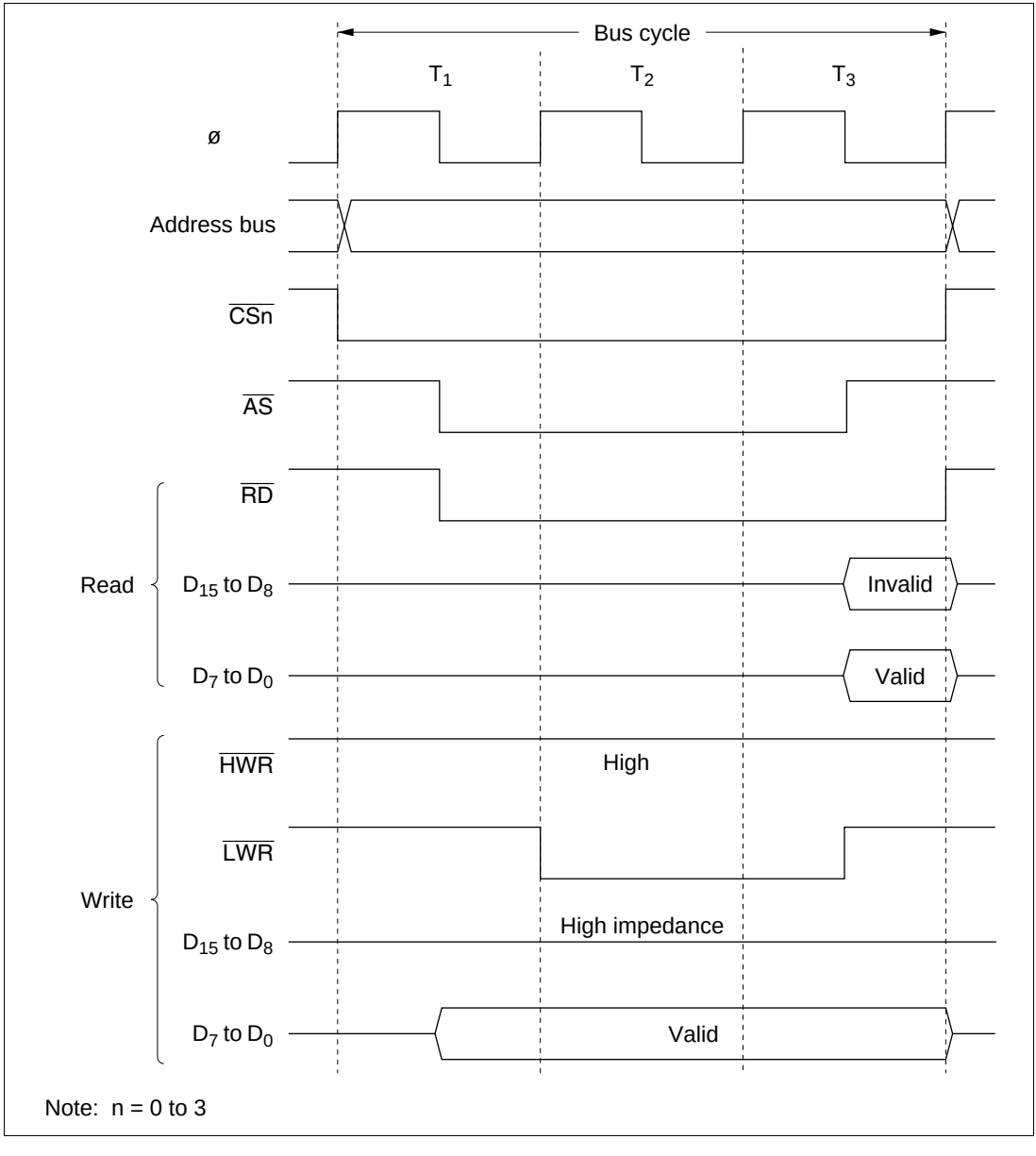



**Figure 6.12 Bus Timing for 16-Bit 3-State Access Space (2) (Odd Address Byte Access)**

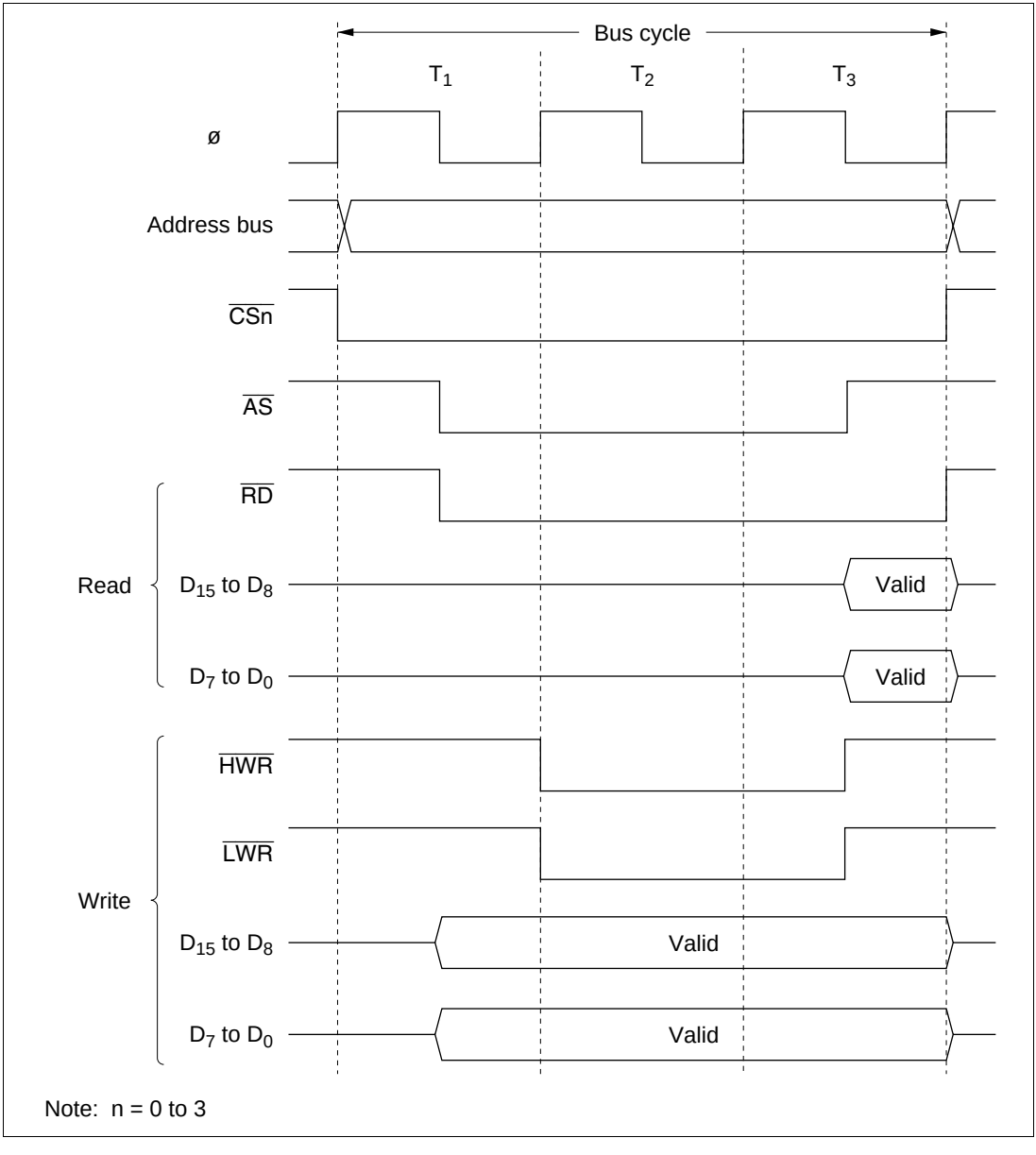



**Figure 6.13 Bus Timing for 16-Bit 3-State Access Space (3) (Word Access)**

### 6.4.5 Wait Control

When accessing external space, the H8S/2345 Series can extend the bus cycle by inserting one or more wait states ($T_w$). There are two ways of inserting wait states: program wait insertion and pin wait insertion using the $\overline{\text{WAIT}}$ pin.

**Program Wait Insertion**

From 0 to 3 wait states can be inserted automatically between the $T_2$ state and $T_3$ state on an individual area basis in 3-state access space, according to the settings of BWCRH and BWCRL.

**Pin Wait Insertion**

Setting the WAITE bit in BCRL to 1 enables wait insertion by means of the $\overline{\text{WAIT}}$ pin. Program wait insertion is first carried out according to the settings in WCRH and WCRL. Then , if the $\overline{\text{WAIT}}$ pin is low at the falling edge of ø in the last $T_2$ or $T_w$ state, a $T_w$ state is inserted. If the $\overline{\text{WAIT}}$ pin is held low, $T_w$ states are inserted until it goes high.

This is useful when inserting four or more $T_w$ states, or when changing the number of $T_w$ states for different external devices.

The WAITE bit setting applies to all areas.

Figure 6.14 shows an example of wait state insertion timing.

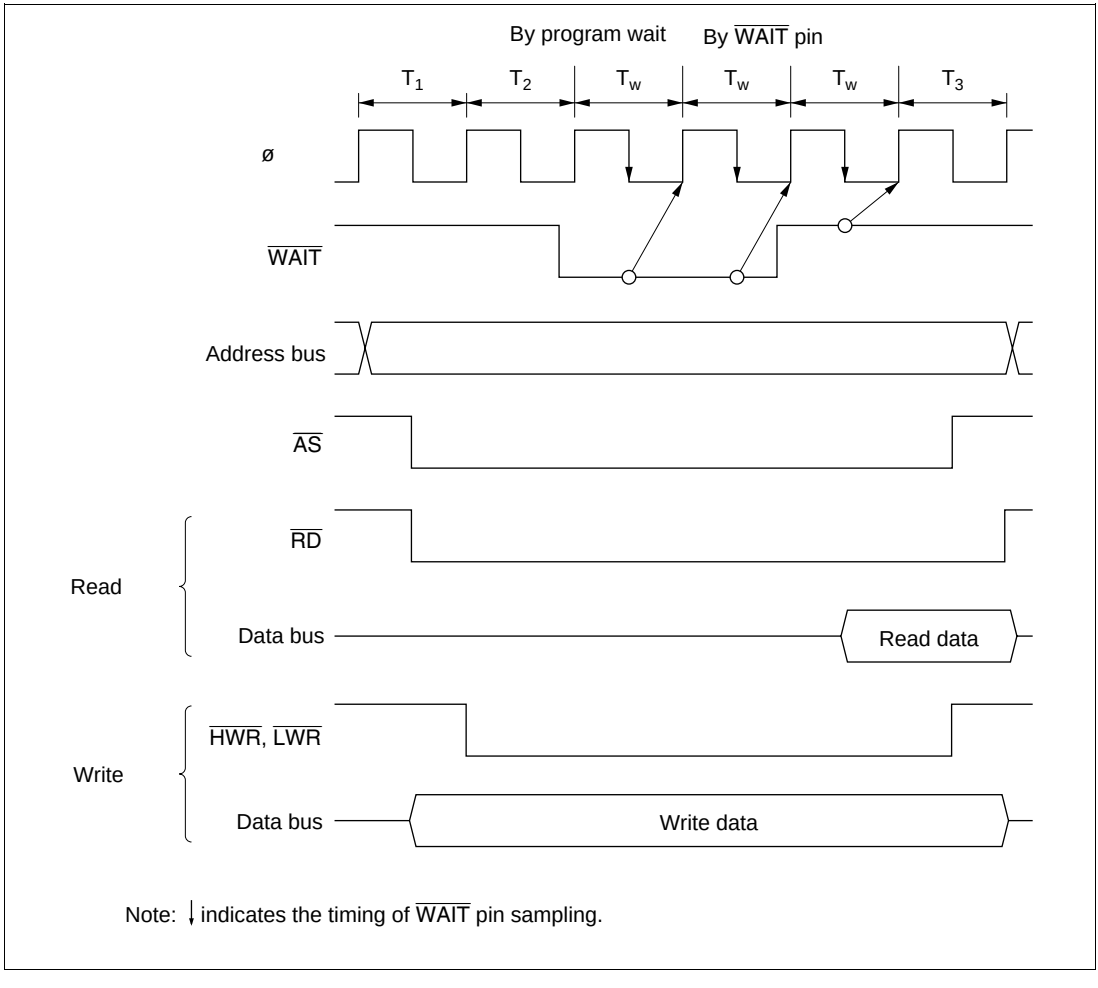



**Figure 6.14 Example of Wait State Insertion Timing**

The settings after a power-on reset are: 3-state access, 3 program wait state insertion, and $\overline{\text{WAIT}}$ input disabled. When a manual reset is performed, the contents of bus controller registers are retained, and the wait control settings remain the same as before the reset.

## 6.5 Burst ROM Interface

### 6.5.1 Overview

With the H8S/2345 Series, external space area 0 can be designated as burst ROM space, and burst ROM interfacing can be performed. The burst ROM space interface enables 16-bit configuration ROM with burst access capability to be accessed at high speed.

Area 0 can be designated as burst ROM space by means of the BRSTRM bit in BCRH. Consecutive burst accesses of a maximum of 4 words or 8 words can be performed for CPU instruction fetches only. One or two states can be selected for burst access.

### 6.5.2 Basic Timing

The number of states in the initial cycle (full access) of the burst ROM interface is in accordance with the setting of the AST0 bit in ASTCR. Also, when the AST0 bit is set to 1, wait state insertion is possible. One or two states can be selected for the burst cycle, according to the setting of the BRSTS1 bit in BCRH. Wait states cannot be inserted. When area 0 is designated as burst ROM space, it becomes 16-bit access space regardless of the setting of the ABW0 bit in ABWCR.

When the BRSTS0 bit in BCRH is cleared to 0, burst access of up to 4 words is performed; when the BRSTS0 bit is set to 1, burst access of up to 8 words is performed.

The basic access timing for burst ROM space is shown in figures 6.15 (a) and (b). The timing shown in figure 6.15 (a) is for the case where the AST0 and BRSTS1 bits are both set to 1, and that in figure 6.15 (b) is for the case where both these bits are cleared to 0.

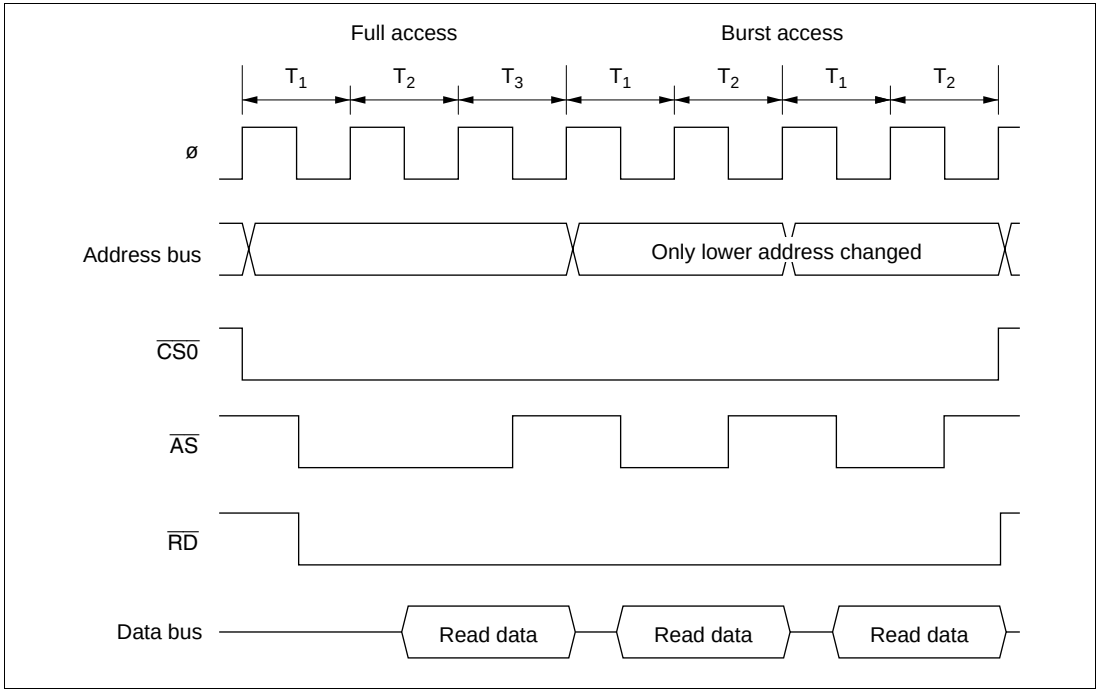



**Figure 6.15 (a) Example of Burst ROM Access Timing (When AST0 = BRSTS1 = 1)**

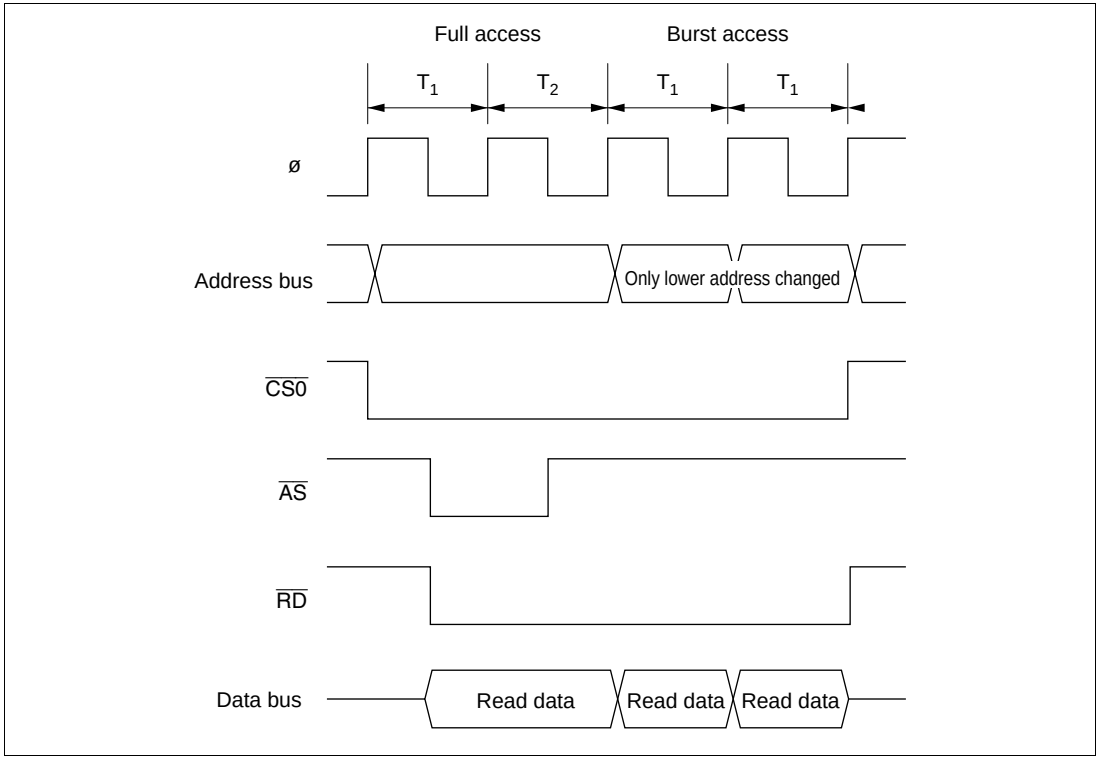



**Figure 6.15 (b) Example of Burst ROM Access Timing (When AST0 = BRSTS1 = 0)**

### 6.5.3 Wait Control

As with the basic bus interface, either program wait insertion or pin wait insertion using the $\overline{\text{WAIT}}$ pin can be used in the initial cycle (full access) of the burst ROM interface. See section 6.4.5, Wait Control.

Wait states cannot be inserted in a burst cycle.

## 6.6 Idle Cycle

### 6.6.1 Operation

When the H8S/2345 Series accesses external space , it can insert a 1-state idle cycle ($T_I$) between bus cycles in the following two cases: (1) when read accesses between different areas occur consecutively, and (2) when a write cycle occurs immediately after a read cycle. By inserting an idle cycle it is possible, for example, to avoid data collisions between ROM, with a long output floating time, and high-speed memory, I/O interfaces, and so on.

**(1) Consecutive Reads between Different Areas**

If consecutive reads between different areas occur while the ICIS1 bit in BCRH is set to 1, an idle cycle is inserted at the start of the second read cycle. This is enabled in advanced mode.

Figure 6.16 shows an example of the operation in this case. In this example, bus cycle A is a read cycle from ROM with a long output floating time, and bus cycle B is a read cycle from SRAM, each being located in a different area. In (a), an idle cycle is not inserted, and a collision occurs in cycle B between the read data from ROM and that from SRAM. In (b), an idle cycle is inserted, and a data collision is prevented.

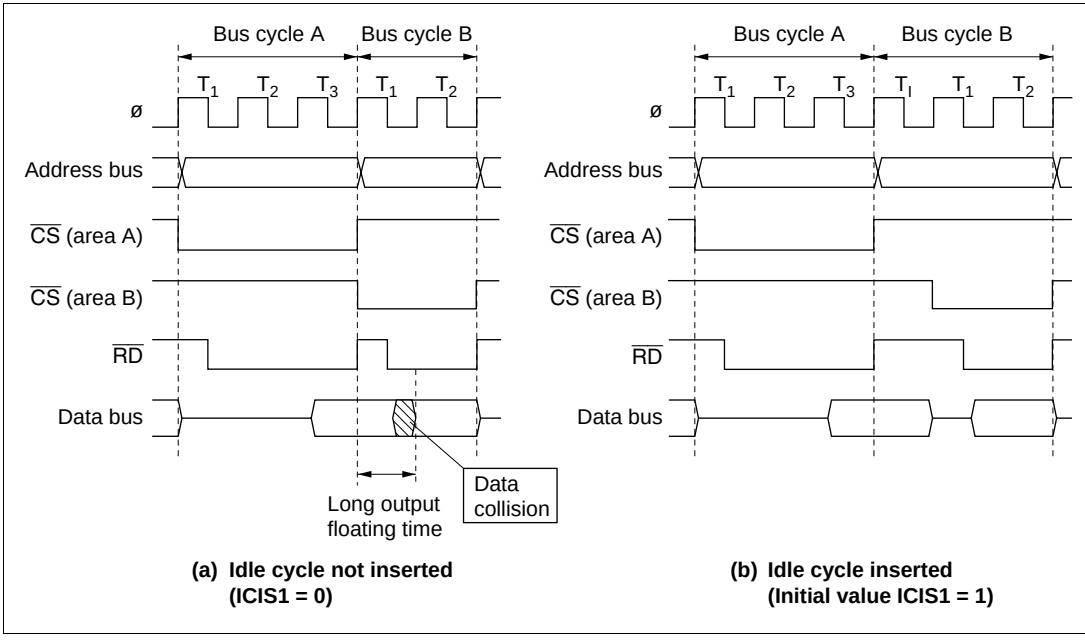



**Figure 6.16 Example of Idle Cycle Operation (1)**

### (2) Write after Read

If an external write occurs after an external read while the ICIS0 bit in BCRH is set to 1, an idle cycle is inserted at the start of the write cycle.

Figure 6.17 shows an example of the operation in this case. In this example, bus cycle A is a read cycle from ROM with a long output floating time, and bus cycle B is a CPU write cycle. In (a), an idle cycle is not inserted, and a collision occurs in cycle B between the read data from ROM and the CPU write data. In (b), an idle cycle is inserted, and a data collision is prevented.

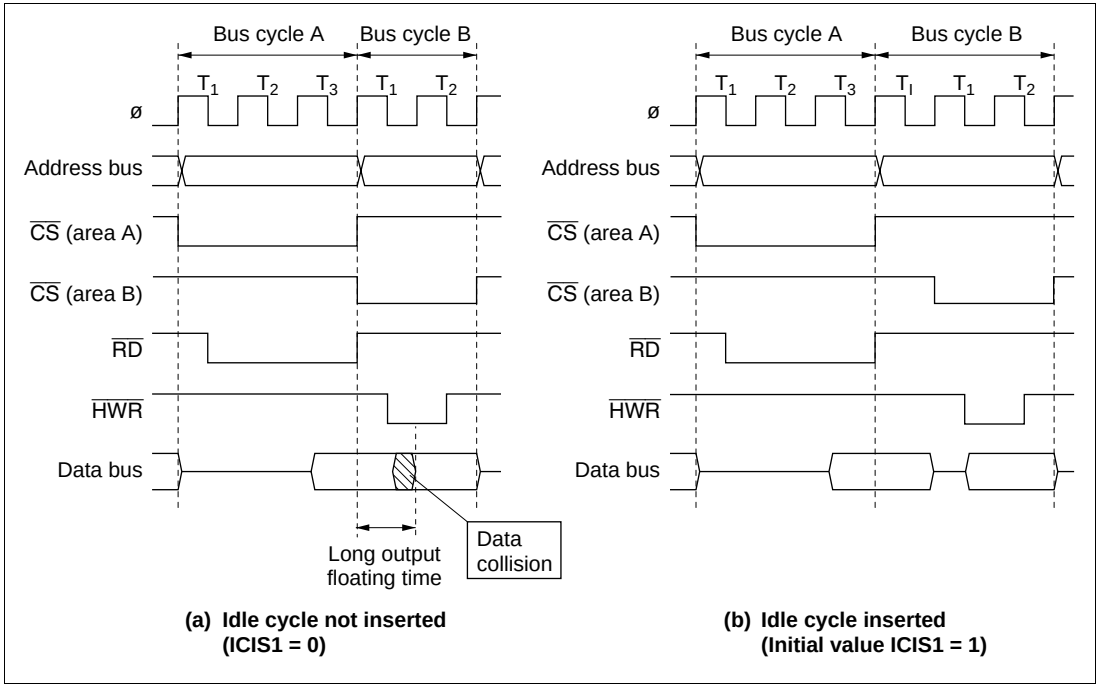



**Figure 6.17 Example of Idle Cycle Operation (2)**

### (3) Relationship between Chip Select ($\overline{CS}$) Signal and Read ($\overline{RD}$) Signal

Depending on the system's load conditions, the $\overline{RD}$ signal may lag behind the $\overline{CS}$ signal. An example is shown in figure 6.18.

In this case, with the setting for no idle cycle insertion (a), there may be a period of overlap between the bus cycle A $\overline{RD}$ signal and the bus cycle B $\overline{CS}$ signal.

Setting idle cycle insertion, as in (b), however, will prevent any overlap between the $\overline{RD}$ and $\overline{CS}$ signals.

In the initial state after reset release, idle cycle insertion (b) is set.

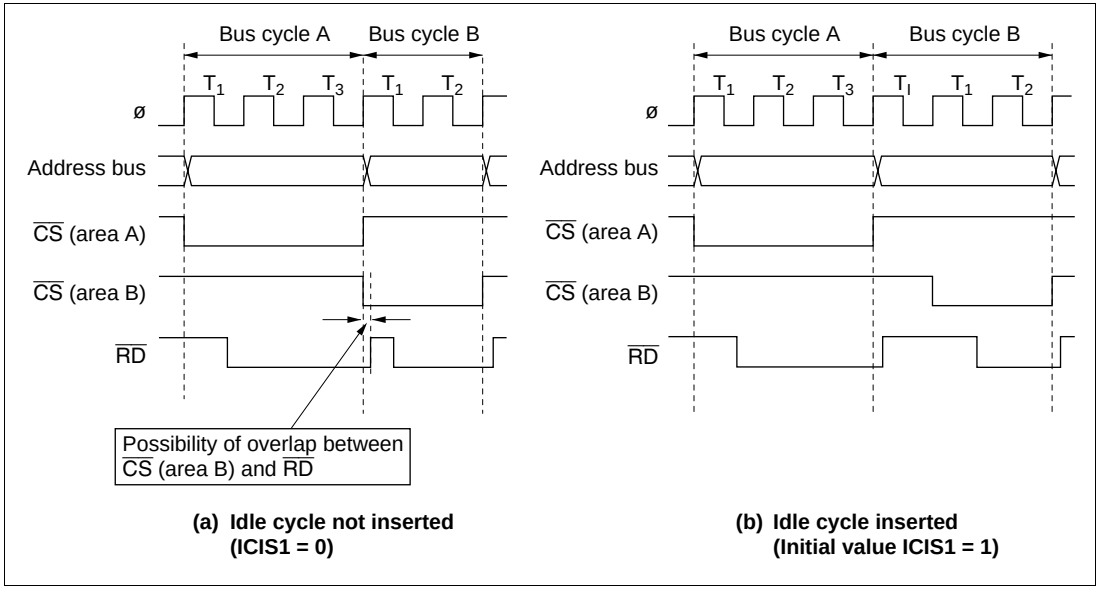



**Figure 6.18 Relationship between Chip Select ($\overline{CS}$) and Read ($\overline{RD}$)**

### 6.6.2 Pin States in Idle Cycle

Table 6.5 shows pin states in an idle cycle.

**Table 6.5 Pin States in Idle Cycle**

| Pins | Pin State |
|---|---|
| $A_{23}$ to $A_0$ | Contents of next bus cycle |
| $D_{15}$ to $D_0$ | High impedance |
| $\overline{CSn}$ | High |
| $\overline{AS}$ | High |
| $\overline{RD}$ | High |
| $\overline{HWR}$ | High |
| $\overline{LWR}$ | High |

## 6.7 Bus Release

### 6.7.1 Overview

The H8S/2345 Series can release the external bus in response to a bus request from an external device. In the external bus released state, the internal bus master continues to operate as long as there is no external access.

### 6.7.2 Operation

In external expansion mode, the bus can be released to an external device by setting the BRLE bit in BCRL to 1. Driving the $\overline{BREQ}$ pin low issues an external bus request to the H8S/2345 Series. When the $\overline{BREQ}$ pin is sampled, at the prescribed timing the $\overline{BACK}$ pin is driven low, and the address bus, data bus, and bus control signals are placed in the high-impedance state, establishing the external bus-released state.

In the external bus released state, an internal bus master can perform accesses using the internal bus. When an internal bus master wants to make an external access, it temporarily defers activation of the bus cycle, and waits for the bus request from the external bus master to be dropped.

When the $\overline{BREQ}$ pin is driven high, the $\overline{BACK}$ pin is driven high at the prescribed timing and the external bus released state is terminated.

In the event of simultaneous external bus release request and external access request generation, the order of priority is as follows:

(High) External bus release > Internal bus master external access (Low)

### 6.7.3 Pin States in External Bus Released State

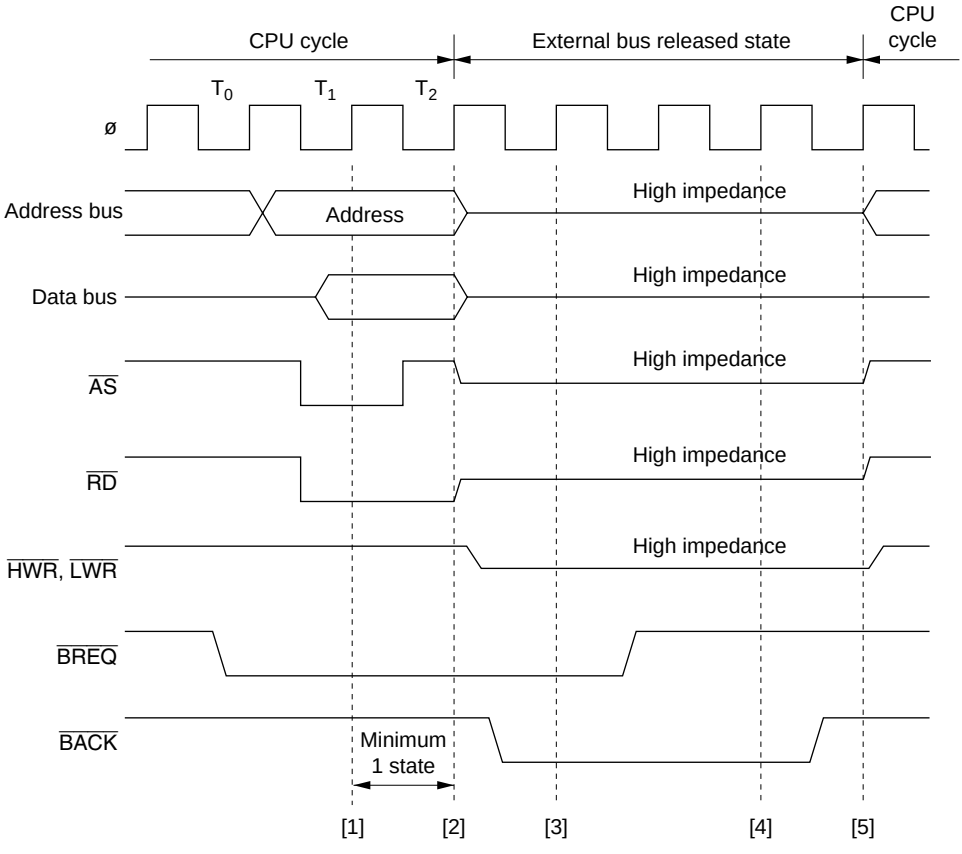
Table 6.6 shows pin states in the external bus released state.

**Table 6.6 Pin States in Bus Released State**

| Pins | Pin State |
|---|---|
| $A_{23}$ to $A_0$ | High impedance |
| $D_{15}$ to $D_0$ | High impedance |
| $\overline{CSn}$ | High impedance |
| $\overline{AS}$ | High impedance |
| $\overline{RD}$ | High impedance |
| $\overline{HWR}$ | High impedance |
| $\overline{LWR}$ | High impedance |

### 6.7.4 Transition Timing

Figure 6.19 shows the timing for transition to the bus-released state.




[1] Low level of $\overline{\text{BREQ}}$ pin is sampled at rise of $T_2$ state.

[2] $\overline{\text{BACK}}$ pin is driven low at end of CPU read cycle, releasing bus to external bus master.

[3] $\overline{\text{BREQ}}$ pin state is still sampled in external bus released state.

[4] High level of $\overline{\text{BREQ}}$ pin is sampled.

[5] $\overline{\text{BACK}}$ pin is driven high, ending bus release cycle.

**Figure 6.19 Bus-Released State Transition Timing**

### 6.7.5 Usage Note

When MSTPCR is set to H'FFFF or H'EFFF and a transition is made to sleep mode, the external bus release function halts. Therefore, MSTPCR should not be set to H'FFFF or H'EFFF if the external bus release function is to be used in sleep mode.

## 6.8 Bus Arbitration

### 6.8.1 Overview

The H8S/2345 Series has a bus arbiter that arbitrates bus master operations.

There are two bus masters, the CPU and DTC, which perform read/write operations when they have possession of the bus. Each bus master requests the bus by means of a bus request signal. The bus arbiter determines priorities at the prescribed timing, and permits use of the bus by means of a bus request acknowledge signal. The selected bus master then takes possession of the bus and begins its operation.

### 6.8.2 Operation

The bus arbiter detects the bus masters' bus request signals, and if the bus is requested, sends a bus request acknowledge signal to the bus master making the request. If there are bus requests from more than one bus master, the bus request acknowledge signal is sent to the one with the highest priority. When a bus master receives the bus request acknowledge signal, it takes possession of the bus until that signal is canceled.

The order of priority of the bus masters is as follows:

(High) DTC > CPU (Low)

An internal bus access by an internal bus master, and external bus release, can be executed in parallel.

In the event of simultaneous external bus release request, and internal bus master external access request generation, the order of priority is as follows:

(High) External bus release > Internal bus master external access (Low)

### 6.8.3 Bus Transfer Timing

Even if a bus request is received from a bus master with a higher priority than that of the bus master that has acquired the bus and is currently operating, the bus is not necessarily transferred immediately. There are specific times at which each bus master can relinquish the bus.

**CPU:** The CPU is the lowest-priority bus master, and if a bus request is received from the DTC, the bus arbiter transfers the bus to the bus master that issued the request. The timing for transfer of the bus is as follows:

- The bus is transferred at a break between bus cycles. However, if a bus cycle is executed in discrete operations, as in the case of a longword-size access, the bus is not transferred between the operations. See Appendix A-5, Bus States During Instruction Execution, for timings at which the bus is not transferred.
- If the CPU is in sleep mode, it transfers the bus immediately.

**DTC:** The DTC sends the bus arbiter a request for the bus when an activation request is generated.

The DTC can release the bus after a vector read, a register information read (3 states), a single data transfer, or a register information write (3 states). It does not release the bus during a register information read (3 states), a single data transfer, or a register information write (3 states).

### 6.8.4 External Bus Release Usage Note

External bus release can be performed on completion of an external bus cycle. The $\overline{RD}$ signal and $\overline{CS0}$ to $\overline{CS3}$ signals remain low until the end of the external bus cycle. Therefore, when external bus release is performed, the $\overline{RD}$ and $\overline{CS0}$ to $\overline{CS3}$ signals may change from the low level to the high-impedance state.

## 6.9 Resets and the Bus Controller

In a power-on reset, the H8S/2345, including the bus controller, enters the reset state at that point, and an executing bus cycle is discontinued.

In a manual reset, the bus controller's registers and internal state are maintained, and an executing external bus cycle is completed. In this case, $\overline{WAIT}$ input is ignored and write data is not guaranteed.

# Section 7 Data Transfer Controller

## 7.1 Overview

The H8S/2345 Series includes a data transfer controller (DTC). The DTC can be activated by an interrupt or software, to transfer data.

### 7.1.1 Features

The features of the DTC are:

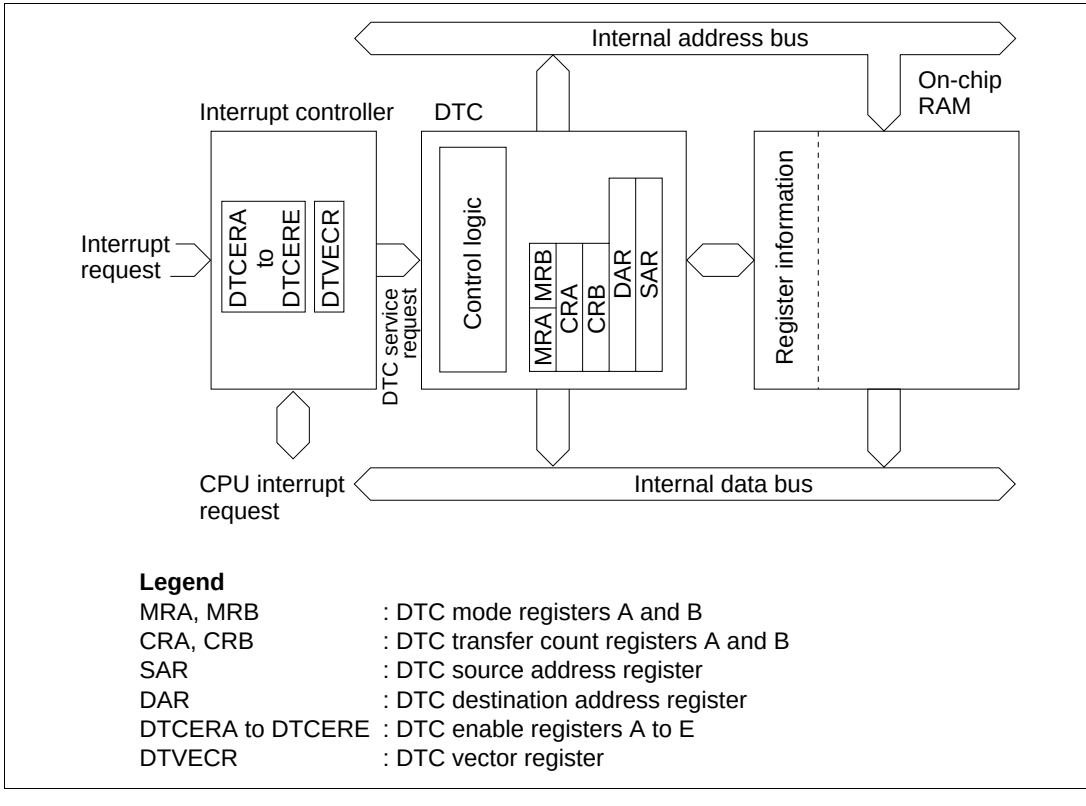
- Transfer possible over any number of channels
  - — Transfer information is stored in memory
  - — One activation source can trigger a number of data transfers (chain transfer)
- Wide range of transfer modes
  - — Normal, repeat, and block transfer modes available
  - — Incrementing, decrementing, and fixing of source and destination addresses can be selected
- Direct specification of 16-Mbyte address space possible
  - — 24-bit transfer source and destination addresses can be specified
- Transfer can be set in byte or word units
- A CPU interrupt can be requested for the interrupt that activated the DTC
  - — An interrupt request can be issued to the CPU after one data transfer ends
  - — An interrupt request can be issued to the CPU after the specified data transfers have completely ended
- Activation by software is possible
- Module stop mode can be set
  - — The initial setting enables DTC registers to be accessed. DTC operation is halted by setting module stop mode.

### 7.1.2 Block Diagram

Figure 7.1 shows a block diagram of the DTC.

The DTC's register information is stored in the on-chip RAM*. A 32-bit bus connects the DTC to the on-chip RAM (1 kbyte), enabling 32-bit/1-state reading and writing of the DTC register information and hence helping to increase processing speed.

Note: * When the DTC is used, the RAME bit in SYSCR must be set to 1.




**Legend**

| | |
|---|---|
| MRA, MRB | : DTC mode registers A and B |
| CRA, CRB | : DTC transfer count registers A and B |
| SAR | : DTC source address register |
| DAR | : DTC destination address register |
| DTCERA to DTCERE | : DTC enable registers A to E |
| DTVECR | : DTC vector register |

**Figure 7.1 Block Diagram of DTC**

### 7.1.3 Register Configuration

Table 7.1 summarizes the DTC registers.

**Table 7.1 DTC Registers**

| Name | Abbreviation | R/W | Initial Value | Address*[1] |
|---|---|---|---|---|
| DTC mode register A | MRA | —*[2] | Undefined | —*[3] |
| DTC mode register B | MRB | —*[2] | Undefined | —*[3] |
| DTC source address register | SAR | —*[2] | Undefined | —*[3] |
| DTC destination address register | DAR | —*[2] | Undefined | —*[3] |
| DTC transfer count register A | CRA | —*[2] | Undefined | —*[3] |
| DTC transfer count register B | CRB | —*[2] | Undefined | —*[3] |
| DTC enable registers | DTCER | R/W | H'00 | H'FF30 to H'FF34 |
| DTC vector register | DTVECR | R/W | H'00 | H'FF37 |
| Module stop control register | MSTPCR | R/W | H'3FFF | H'FF3C |

Notes: 1. Lower 16 bits of the address.
2. Registers within the DTC cannot be read or written to directly.
3. Register information is located in on-chip RAM addresses H'F800 to H'FBFF. It cannot be located in external space. When the DTC is used, do not clear the RAME bit in SYSCR to 0.

## 7.2 Register Descriptions

### 7.2.1 DTC Mode Register A (MRA)

MRA is an 8-bit register that controls the DTC operating mode.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | SM1 | SM0 | DM1 | DM0 | MD1 | MD0 | DTS | Sz |
| Initial value | : | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined |
| R/W | : | — | — | — | — | — | — | — | — |

**Bits 7 and 6—Source Address Mode 1 and 0 (SM1, SM0):** These bits specify whether SAR is to be incremented, decremented, or left fixed after a data transfer.

| Bit 7 | Bit 6 | |
|---|---|---|
| **SM1** | **SM0** | **Description** |
| 0 | — | SAR is fixed |
| 1 | 0 | SAR is incremented after a transfer<br>(by +1 when Sz = 0; by +2 when Sz = 1) |
| | 1 | SAR is decremented after a transfer<br>(by –1 when Sz = 0; by –2 when Sz = 1) |

**Bits 5 and 4—Destination Address Mode 1 and 0 (DM1, DM0):** These bits specify whether DAR is to be incremented, decremented, or left fixed after a data transfer.

| Bit 5 | Bit 4 | |
|---|---|---|
| **DM1** | **DM0** | **Description** |
| 0 | — | DAR is fixed |
| 1 | 0 | DAR is incremented after a transfer<br>(by +1 when Sz = 0; by +2 when Sz = 1) |
| | 1 | DAR is decremented after a transfer<br>(by –1 when Sz = 0; by –2 when Sz = 1) |

**Bits 3 and 2—DTC Mode (MD1, MD0):** These bits specify the DTC transfer mode.

| Bit 3 | Bit 2 | |
|---|---|---|
| **MD1** | **MD0** | **Description** |
| 0 | 0 | Normal mode |
| | 1 | Repeat mode |
| 1 | 0 | Block transfer mode |
| | 1 | — |

**Bit 1—DTC Transfer Mode Select (DTS):** Specifies whether the source side or the destination side is set to be a repeat area or block area, in repeat mode or block transfer mode.

| Bit 1 | |
|---|---|
| **DTS** | **Description** |
| 0 | Destination side is repeat area or block area |
| 1 | Source side is repeat area or block area |

**Bit 0—DTC Data Transfer Size (Sz):** Specifies the size of data to be transferred.

| Bit 0 | |
|---|---|
| **Sz** | **Description** |
| 0 | Byte-size transfer |
| 1 | Word-size transfer |

### 7.2.2 DTC Mode Register B (MRB)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | CHNE | DISEL | — | — | — | — | — | — |
| Initial value: | | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined |
| R/W | : | — | — | — | — | — | — | — | — |

MRB is an 8-bit register that controls the DTC operating mode.

**Bit 7—DTC Chain Transfer Enable (CHNE):** Specifies chain transfer. With chain transfer, a number of data transfers can be performed consecutively in response to a single transfer request.

In data transfer with CHNE set to 1, determination of the end of the specified number of transfers, clearing of the interrupt source flag, and clearing of DTCER is not performed.

**Bit 7**

| CHNE | Description |
|---|---|
| 0 | End of DTC data transfer (activation waiting state is entered) |
| 1 | DTC chain transfer (new register information is read, then data is transferred) |

**Bit 6—DTC Interrupt Select (DISEL):** Specifies whether interrupt requests to the CPU are disabled or enabled after a data transfer.

**Bit 6**

| DISEL | Description |
|---|---|
| 0 | After a data transfer ends, the CPU interrupt is disabled unless the transfer counter is 0 (the DTC clears the interrupt source flag of the activating interrupt to 0) |
| 1 | After a data transfer ends, the CPU interrupt is enabled (the DTC does not clear the interrupt source flag of the activating interrupt to 0) |

**Bits 5 to 0—Reserved:** These bits have no effect on DTC operation in the H8S/2345 Series, and should always be written with 0 in a write.

### 7.2.3 DTC Source Address Register (SAR)

| Bit | : | 23 | 22 | 21 | 20 | 19 | - - - | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | - - - | | | | | |
| Initial value: | | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | - - - | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined |
| R/W | : | — | — | — | — | — | - - - | — | — | — | — | — |

SAR is a 24-bit register that designates the source address of data to be transferred by the DTC. For word-size transfer, specify an even source address.

### 7.2.4 DTC Destination Address Register (DAR)

| Bit | : | 23 | 22 | 21 | 20 | 19 | - - - | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | - - - | | | | | |
| Initial value | : | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | - - - | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined |
| R/W | : | — | — | — | — | — | - - - | — | — | — | — | — |

DAR is a 24-bit register that designates the destination address of data to be transferred by the DTC. For word-size transfer, specify an even destination address.

### 7.2.5 DTC Transfer Count Register A (CRA)

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | |
| Initial value: | | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined |
| R/W | : | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| | | ← CRAH → | | | | | | | | ← CRAL → | | | | | | | |

CRA is a 16-bit register that designates the number of times data is to be transferred by the DTC.

In normal mode, the entire CRA functions as a 16-bit transfer counter (1 to 65536). It is decremented by 1 every time data is transferred, and transfer ends when the count reaches H'0000.

In repeat mode or block transfer mode, the CRA is divided into two parts: the upper 8 bits (CRAH) and the lower 8 bits (CRAL). CRAH holds the number of transfers while CRAL functions as an 8-bit transfer counter (1 to 256). CRAL is decremented by 1 every time data is transferred, and the contents of CRAH are sent when the count reaches H'00. This operation is repeated.

### 7.2.6 DTC Transfer Count Register B (CRB)

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | |
| Initial value: | | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined |
| R/W | : | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |

CRB is a 16-bit register that designates the number of times data is to be transferred by the DTC in block transfer mode. It functions as a 16-bit transfer counter (1 to 65536) that is decremented by 1 every time data is transferred, and transfer ends when the count reaches H'0000.

### 7.2.7 DTC Enable Registers (DTCER)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | DTCE7 | DTCE6 | DTCE5 | DTCE4 | DTCE3 | DTCE2 | DTCE1 | DTCE0 |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The DTC enable registers comprise five 8-bit readable/writable registers, DTCERA to DTCERE, with bits corresponding to the interrupt sources that can activate the DTC. These bits enable or disable DTC service for the corresponding interrupt sources.

The DTC enable registers are initialized to H'00 by a reset and in hardware standby mode.

A DTCE bit can be set for each interrupt source that can activate the DTC. The correspondence between interrupt sources and DTCE bits is shown in table 7.4, together with the vector number generated for each interrupt controller.

For DTCE bit setting, read/write operations must be performed using bit-manipulation instructions such as BSET and BCLR. For the initial setting only, however, when multiple activation sources are set at one time, it is possible to disable interrupts and write after executing a dummy read on the relevant register.

**Bit n—DTC Activation Enable (DTCEn)**

| Bit n<br>DTCEn | Description | |
|---|---|---|
| 0 | DTC activation by this interrupt is disabled<br>[Clearing conditions]<br>• When the DISEL bit is 1 and the data transfer has ended<br>• When the specified number of transfers have ended | (Initial value) |
| 1 | DTC activation by this interrupt is enabled<br>[Holding condition]<br>When the DISEL bit is 0 and the specified number of transfers have not ended | |

(n = 7 to 0)

A DTCE bit can be set for each interrupt source that can activate the DTC. The correspondence between interrupt sources and DTCE bits is shown in table 7.4, together with the vector number generated for each interrupt controller.

### 7.2.8 DTC Vector Register (DTVECR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | SWDTE | DTVEC6 | DTVEC5 | DTVEC4 | DTVEC3 | DTVEC2 | DTVEC1 | DTVEC0 |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/(W)* | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note: * A value of 1 can always be written to the SWDTE bit, but 0 can only be written after 1 is read.

DTVECR is an 8-bit readable/writable register that enables or disables DTC activation by software, and sets a vector number for the software activation interrupt.

DTVECR is initialized to H'00 by a reset and in hardware standby mode.

**Bit 7—DTC Software Activation Enable (SWDTE):** Enables or disables DTC activation by software.

When clearing the SWDTE bit to 0 by software, write 0 to SWDTE after reading SWDTE set to 1.

**Bit 7**

| SWDTE | Description | |
|---|---|---|
| 0 | DTC software activation is disabled | (Initial value) |
| | [Clearing condition] | |
| | When the DISEL bit is 0 and the specified number of transfers have not ended | |
| 1 | DTC software activation is enabled | |
| | [Holding conditions] | |
| | • When the DISEL bit is 1 and data transfer has ended | |
| | • When the specified number of transfers have ended | |
| | • During data transfer due to software activation | |

**Bits 6 to 0—DTC Software Activation Vectors 6 to 0 (DTVEC6 to DTVEC0):** These bits specify a vector number for DTC software activation.

The vector address is expressed as H'0400 + ((vector number) << 1). <<1 indicates a one-bit left-shift. For example, when DTVEC6 to DTVEC0 = H'10, the vector address is H'0420.

### 7.2.9 Module Stop Control Register (MSTPCR)

| | | MSTPCRH | | | | | | | | MSTPCRL | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | | |
| Initial value | : | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the MSTP14 bit in MSTPCR is set to 1, the DTC operation stops at the end of the bus cycle and a transition is made to module stop mode. However, 1 cannot be written in the MSTP14 bit while the DTC is operating. For details, see section 19.5, Module Stop Mode.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 14—Module Stop (MSTP14):** Specifies the DTC module stop mode.

**Bit 14**

| MSTP14 | Description | |
|---|---|---|
| 0 | DTC module stop mode cleared | (Initial value) |
| 1 | DTC module stop mode set | |

## 7.3 Operation

### 7.3.1 Overview

When activated, the DTC reads register information that is already stored in memory and transfers data on the basis of that register information. After the data transfer, it writes updated register information back to memory. Pre-storage of register information in memory makes it possible to transfer data over any required number of channels. Setting the CHNE bit to 1 makes it possible to perform a number of transfers with a single activation.

Figure 7.2 shows a flowchart of DTC operation.

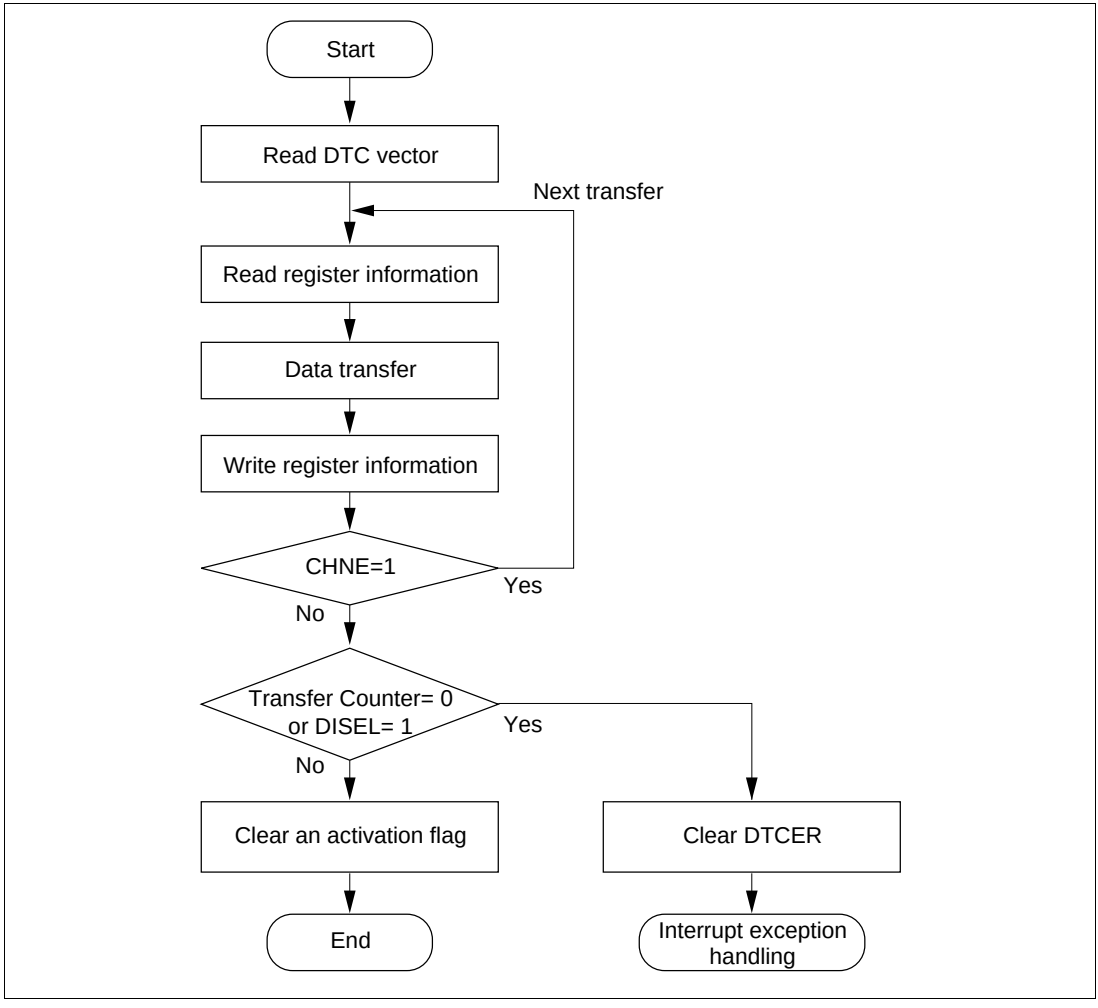



**Figure 7.2 Flowchart of DTC Operation**

The DTC transfer mode can be normal mode, repeat mode, or block transfer mode.

The 24-bit SAR designates the DTC transfer source address and the 24-bit DAR designates the transfer destination address. After each transfer, SAR and DAR are independently incremented, decremented, or left fixed.

Table 7.2 outlines the functions of the DTC.

**Table 7.2 DTC Functions**

| | | Address Registers | |
|---|---|---|---|
| **Transfer Mode** | **Activation Source** | **Transfer Source** | **Transfer Destination** |
| • Normal mode<br>— One transfer request transfers one byte or one word<br>— Memory addresses are incremented or decremented by 1 or 2<br>— Up to 65,536 transfers possible<br>• Repeat mode<br>— One transfer request transfers one byte or one word<br>— Memory addresses are incremented or decremented by 1 or 2<br>— After the specified number of transfers (1 to 256), the initial state resumes and operation continues<br>• Block transfer mode<br>— One transfer request transfers a block of the specified size<br>— Block size is from 1 to 256 bytes or words<br>— Up to 65,536 transfers possible<br>— A block area can be designated at either the source or destination | • IRQ<br>• TPU TGI<br>• 8-bit timer CMI<br>• SCI TXI or RXI<br>• A/D converter ADI<br>• Software | 24 bits | 24 bits |

### 7.3.2 Activation Sources

The DTC operates when activated by an interrupt or by a write to DTVECR by software. An interrupt request can be directed to the CPU or DTC, as designated by the corresponding DTCER bit. An interrupt becomes a DTC activation source when the corresponding bit is set to 1, and a CPU interrupt source when the bit is cleared to 0.

At the end of a data transfer (or the last consecutive transfer in the case of chain transfer), the activation source or corresponding DTCER bit is cleared. Table 7.3 shows activation source and DTCER clearance. The activation source flag, in the case of RXI0, for example, is the RDRF flag of SCI0.

**Table 7.3 Activation Source and DTCER Clearance**

| Activation Source | When the DISEL Bit Is 0 and the Specified Number of Transfers Have Not Ended | When the DISEL Bit Is 1, or when the Specified Number of Transfers Have Ended |
|---|---|---|
| Software activation | The SWDTE bit is cleared to 0 | The SWDTE bit remains set to 1<br>An interrupt is issued to the CPU |
| Interrupt activation | The corresponding DTCER bit remains set to 1<br>The activation source flag is cleared to 0 | The corresponding DTCER bit is cleared to 0<br>The activation source flag remains set to 1<br>A request is issued to the CPU for the activation source interrupt |

Figure 7.3 shows a block diagram of activation source control. For details see section 5, Interrupt Controller.

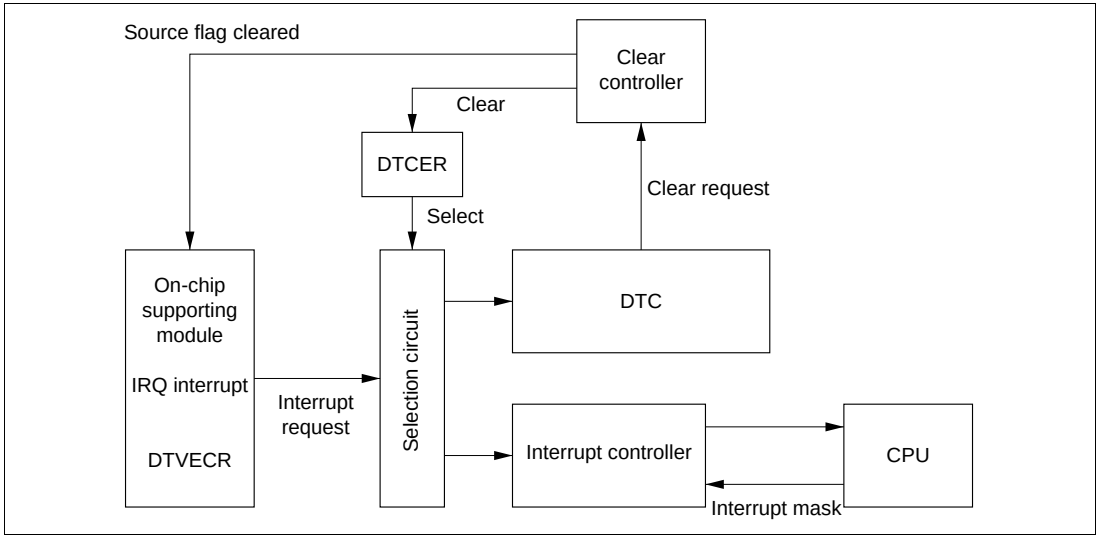



**Figure 7.3 Block Diagram of DTC Activation Source Control**

When an interrupt has been designated a DTC activation source, existing CPU mask level and interrupt controller priorities have no effect. If there is more than one activation source at the same time, the DTC operates in accordance with the default priorities.

### 7.3.3 DTC Vector Table

Figure 7.4 shows the correspondence between DTC vector addresses and register information.

Table 7.4 shows the correspondence between activation, vector addresses, and DTCER bits. When the DTC is activated by software, the vector address is obtained from: H'0400 + (DTVECR[6:0] << 1) (where << 1 indicates a 1-bit left shift). For example, if DTVECR is H'10, the vector address is H'0420.

The DTC reads the start address of the register information from the vector address set for each activation source, and then reads the register information from that start address. The register information can be placed at predetermined addresses in the on-chip RAM. The start address of the register information should be an integral multiple of four.

The configuration of the vector address is the same in both normal and advanced modes, a 2-byte unit being used in both cases. These two bytes specify the lower bits of the address in the on-chip RAM.

**Table 7.4 Interrupt Sources, DTC Vector Addresses, and Corresponding DTCEs**

| Interrupt Source | Origin of Interrupt Source | Vector Number | Vector Address | DTCE* | Priority |
|---|---|---|---|---|---|
| Write to DTVECR | Software | DTVECR | H'0400+ (DTVECR [6:0] <<1) | — | High |
| IRQ0 | External pin | 16 | H'0420 | DTCEA7 | |
| IRQ1 | | 17 | H'0422 | DTCEA6 | |
| IRQ2 | | 18 | H'0424 | DTCEA5 | |
| IRQ3 | | 19 | H'0426 | DTCEA4 | |
| IRQ4 | | 20 | H'0428 | DTCEA3 | |
| IRQ5 | | 21 | H'042A | DTCEA2 | |
| IRQ6 | | 22 | H'042C | DTCEA1 | |
| IRQ7 | | 23 | H'042E | DTCEA0 | |
| ADI (A/D conversion end) | A/D | 28 | H'0438 | DTCEB6 | |
| TGI0A (GR0A compare match/ input capture) | TPU channel 0 | 32 | H'0440 | DTCEB5 | |
| TGI0B (GR0B compare match/ input capture) | | 33 | H'0442 | DTCEB4 | |
| TGI0C (GR0C compare match/ input capture) | | 34 | H'0444 | DTCEB3 | |
| TGI0D (GR0D compare match/ input capture) | | 35 | H'0446 | DTCEB2 | |
| TGI1A (GR1A compare match/ input capture) | TPU channel 1 | 40 | H'0450 | DTCEB1 | |
| TGI1B (GR1B compare match/ input capture) | | 41 | H'0452 | DTCEB0 | |
| TGI2A (GR2A compare match/ input capture) | TPU channel 2 | 44 | H'0458 | DTCEC7 | |
| TGI2B (GR2B compare match/ input capture) | | 45 | H'045A | DTCEC6 | Low |

Note: * DTCE bits with no corresponding interrupt are reserved, and should be written with 0.

**Table 7.4 Interrupt Sources, DTC Vector Addresses, and Corresponding DTCEs (cont)**

| Interrupt Source | Origin of Interrupt Source | Vector Number | Vector Address | DTCE | Priority |
|---|---|---|---|---|---|
| TGI3A (GR3A compare match/ input capture) | TPU channel 3 | 48 | H'0460 | DTCEC5 | High |
| TGI3B (GR3B compare match/ input capture) | | 49 | H'0462 | DTCEC4 | |
| TGI3C (GR3C compare match/ input capture) | | 50 | H'0464 | DTCEC3 | |
| TGI3D (GR3D compare match/ input capture) | | 51 | H'0466 | DTCEC2 | |
| TGI4A (GR4A compare match/ input capture) | TPU channel 4 | 56 | H'0470 | DTCEC1 | |
| TGI4B (GR4B compare match/ input capture) | | 57 | H'0472 | DTCEC0 | |
| TGI5A (GR5A compare match/ input capture) | TPU channel 5 | 60 | H'0478 | DTCED5 | |
| TGI5B (GR5B compare match/ input capture) | | 61 | H'047A | DTCED4 | |
| CMIA0 | 8-bit timer channel 0 | 64 | H'0480 | DTCED3 | |
| CMIB0 | | 65 | H'0482 | DTCED2 | |
| CMIA1 | 8-bit timer channel 1 | 68 | H'0488 | DTCED1 | |
| CMIB1 | | 69 | H'048A | DTCED0 | |
| RXI0 (reception complete 0) | SCI channel 0 | 81 | H'04A2 | DTCEE3 | |
| TXI0 (transmit data empty 0) | | 82 | H'04A4 | DTCEE2 | |
| RXI1 (reception complete 1) | SCI channel 1 | 85 | H'04AA | DTCEE1 | |
| TXI1 (transmit data empty 1) | | 86 | H'04AC | DTCEE0 | Low |

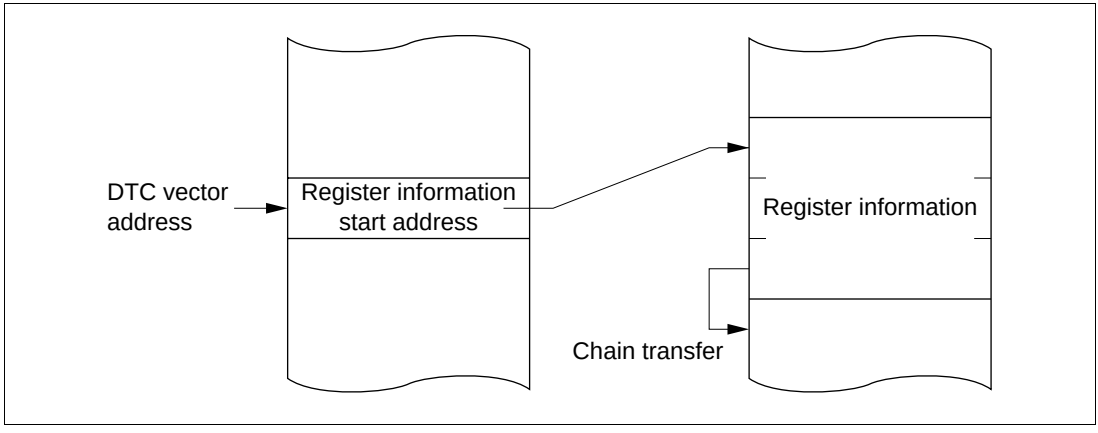



**Figure 7.4 Correspondence between DTC Vector Address and Register Information**

### 7.3.4 Location of Register Information in Address Space

Figure 7.5 shows how the register information should be located in the address space.

Locate the MRA, SAR, MRB, DAR, CRA, and CRB registers, in that order, from the start address of the register information (contents of the vector address). In the case of chain transfer, register information should be located in consecutive areas.

Locate the register information in the on-chip RAM (addresses: H'FFF800 to H'FFFBFF).

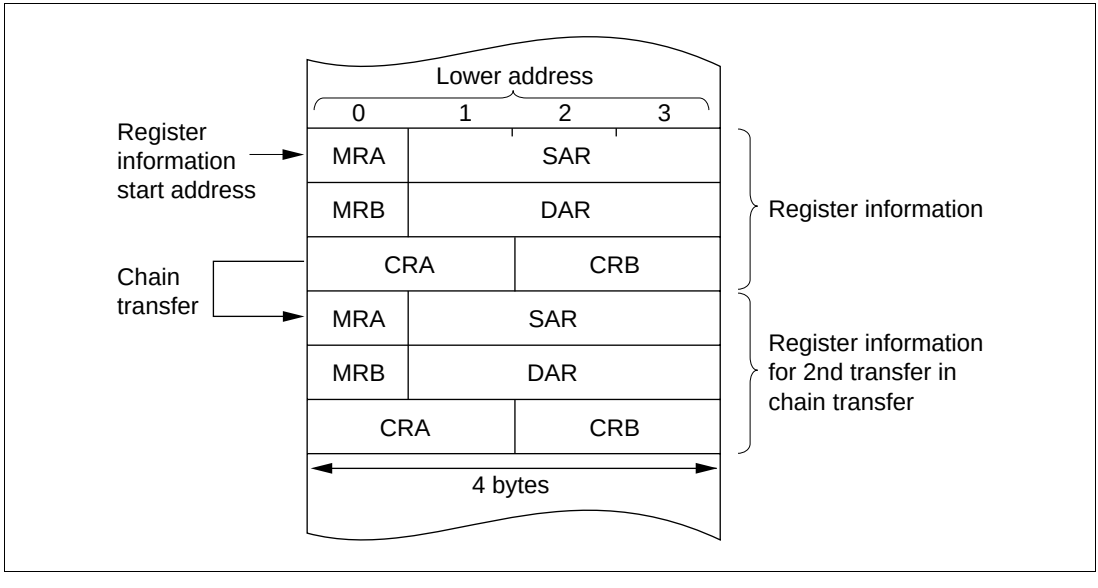



**Figure 7.5 Location of Register Information in Address Space**

### 7.3.5 Normal Mode

In normal mode, one operation transfers one byte or one word of data.

From 1 to 65,536 transfers can be specified. Once the specified number of transfers have ended, a CPU interrupt can be requested.

Table 7.5 lists the register information in normal mode and figure 7.6 shows memory mapping in normal mode.

**Table 7.5 Register Information in Normal Mode**

| Name | Abbreviation | Function |
|---|---|---|
| DTC source address register | SAR | Designates source address |
| DTC destination address register | DAR | Designates destination address |
| DTC transfer count register A | CRA | Designates transfer count |
| DTC transfer count register B | CRB | Not used |

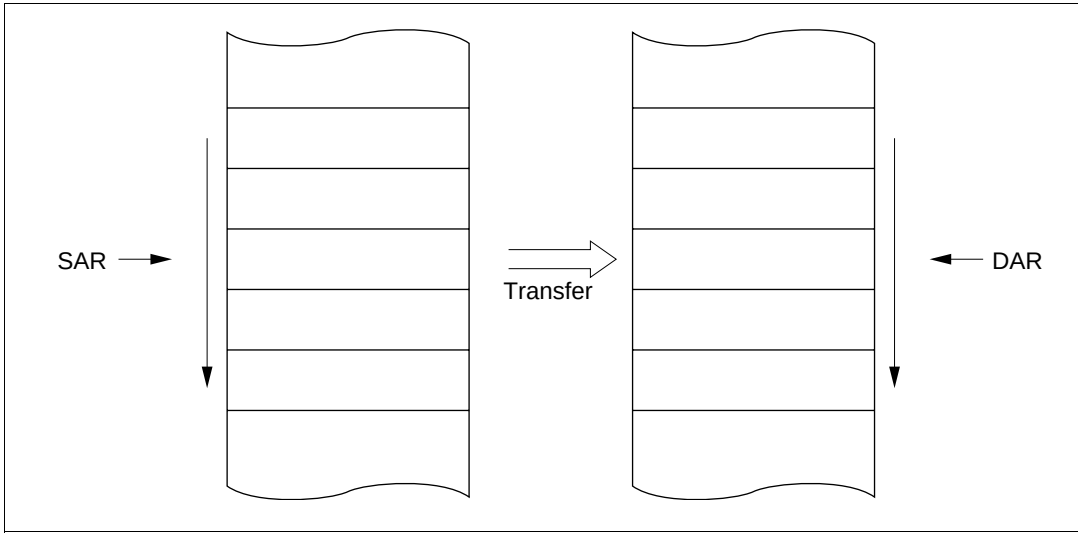



**Figure 7.6 Memory Mapping in Normal Mode**

### 7.3.6 Repeat Mode

In repeat mode, one operation transfers one byte or one word of data.

From 1 to 256 transfers can be specified. Once the specified number of transfers have ended, the initial state of the transfer counter and the address register specified as the repeat area is restored, and transfer is repeated. In repeat mode the transfer counter value does not reach H'00, and therefore CPU interrupts cannot be requested when DISEL = 0.

Table 7.6 lists the register information in repeat mode and figure 7.7 shows memory mapping in repeat mode.

**Table 7.6 Register Information in Repeat Mode**

| Name | Abbreviation | Function |
|---|---|---|
| DTC source address register | SAR | Designates source address |
| DTC destination address register | DAR | Designates destination address |
| DTC transfer count register AH | CRAH | Holds number of transfers |
| DTC transfer count register AL | CRAL | Designates transfer count (8 bits × 2) |
| DTC transfer count register B | CRB | Not used |

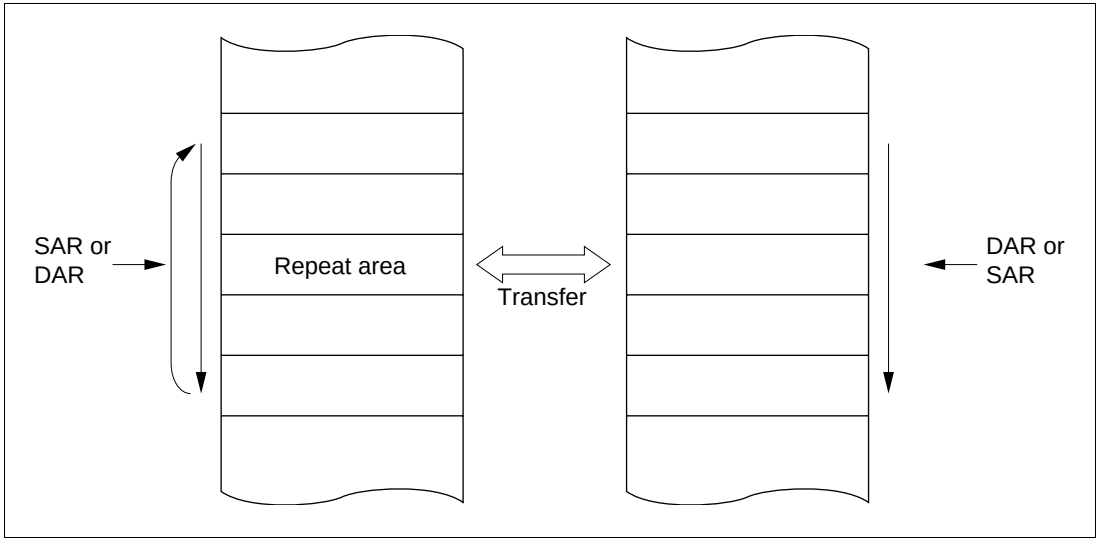



**Figure 7.7 Memory Mapping in Repeat Mode**

### 7.3.7 Block Transfer Mode

In block transfer mode, one operation transfers one block of data.

The block size is 1 to 256. When the transfer of one block ends, the initial state of the block size counter and the address register specified as the block area is restored. The other address register is then incremented, decremented, or left fixed.

From 1 to 65,536 transfers can be specified. Once the specified number of transfers have ended, a CPU interrupt is requested.

Table 7.7 lists the register information in block transfer mode and figure 7.8 shows memory mapping in block transfer mode.

**Table 7.7 Register Information in Block Transfer Mode**

| Name | Abbreviation | Function |
|---|---|---|
| DTC source address register | SAR | Designates transfer source address |
| DTC destination address register | DAR | Designates destination address |
| DTC transfer count register AH | CRAH | Holds block size |
| DTC transfer count register AL | CRAL | Designates block size count |
| DTC transfer count register B | CRB | Transfer count |

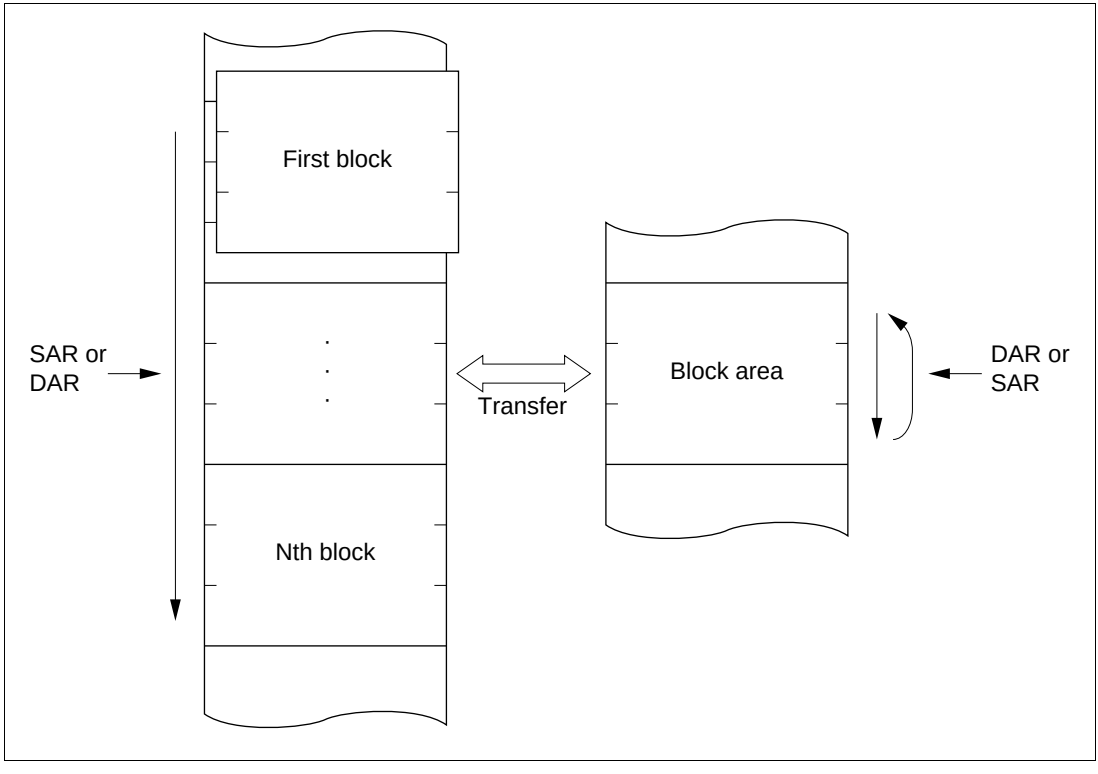



**Figure 7.8 Memory Mapping in Block Transfer Mode**

### 7.3.8 Chain Transfer

Setting the CHNE bit to 1 enables a number of data transfers to be performed consecutively in response to a single transfer request. SAR, DAR, CRA, CRB, MRA, and MRB, which define data transfers, can be set independently.

Figure 7.9 shows the memory map for chain transfer.

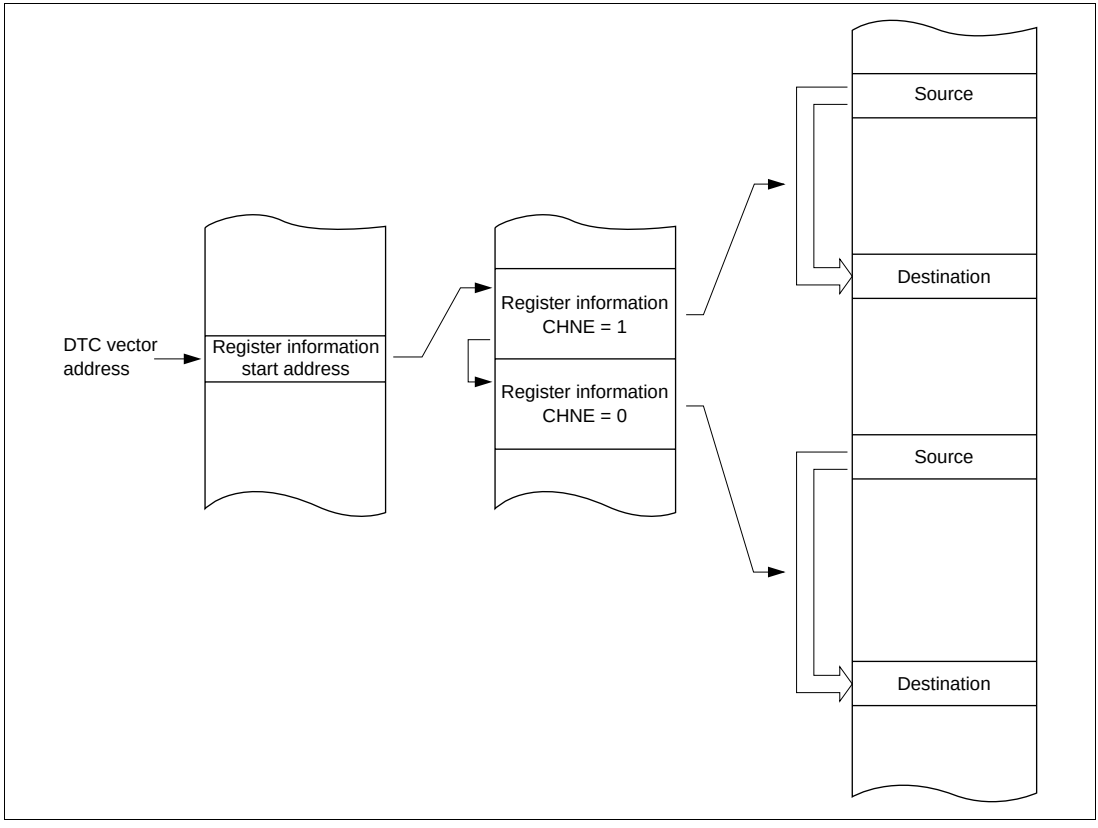



**Figure 7.9 Chain Transfer Memory Map**

In the case of transfer with CHNE set to 1, an interrupt request to the CPU is not generated at the end of the specified number of transfers or by setting of the DISEL bit to 1, and the interrupt source flag for the activation source is not affected.

### 7.3.9 Operation Timing

Figures 7.10 to 7.12 show an example of DTC operation timing.

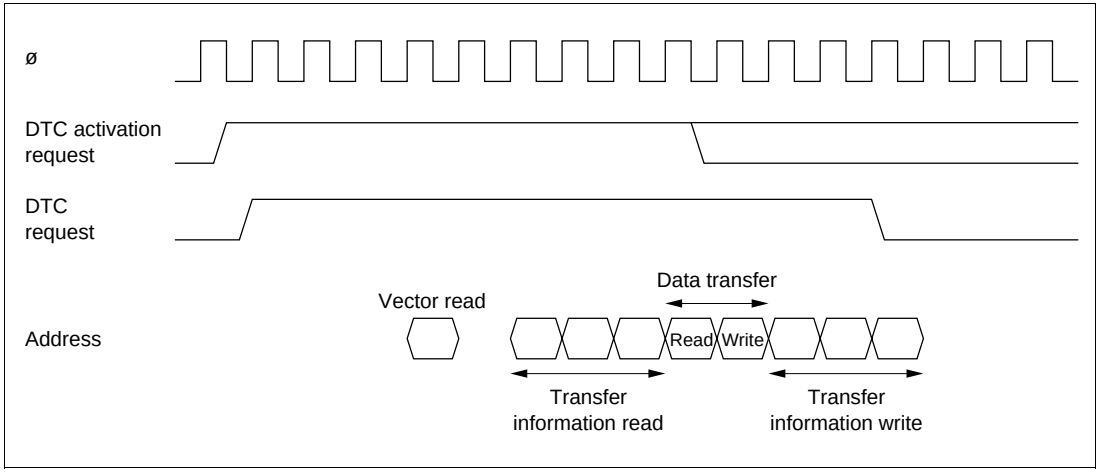



**Figure 7.10 DTC Operation Timing (Example in Normal Mode or Repeat Mode)**

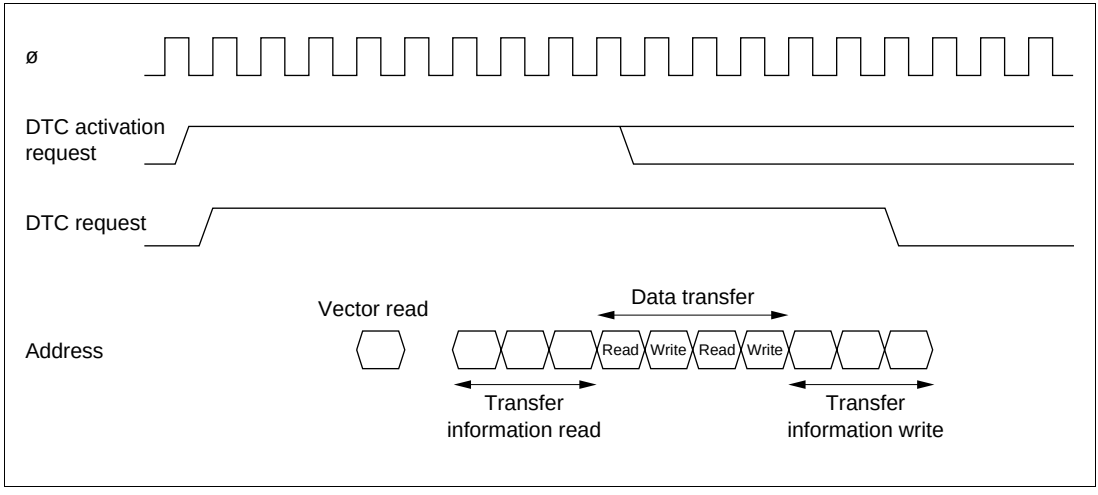



**Figure 7.11 DTC Operation Timing (Example of Block Transfer Mode, with Block Size of 2)**

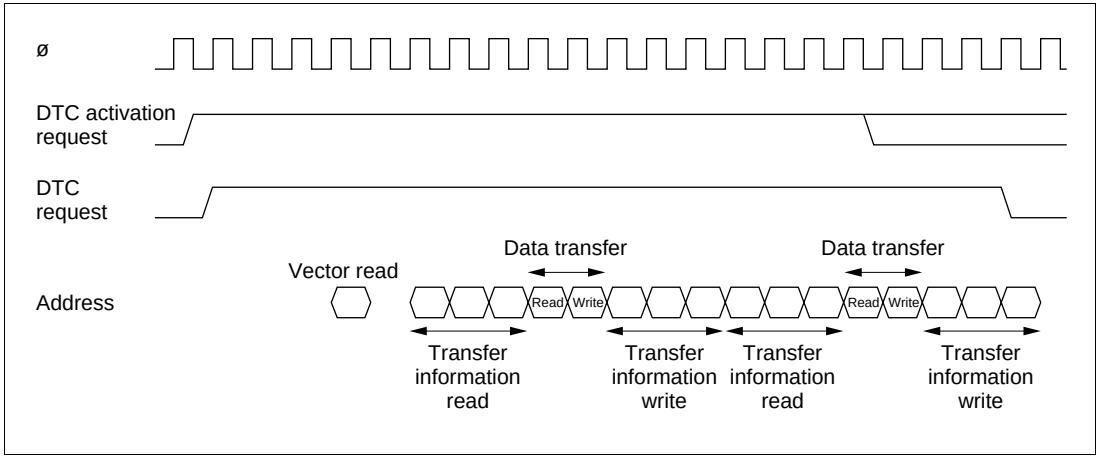



**Figure 7.12 DTC Operation Timing (Example of Chain Transfer)**

### 7.3.10 Number of DTC Execution States

Table 7.8 lists execution statuses for a single DTC data transfer, and table 7.9 shows the number of states required for each execution status.

**Table 7.8 DTC Execution Statuses**

| Mode | Vector Read I | Register Information Read/Write J | Data Read K | Data Write L | Internal Operations M |
|---|---|---|---|---|---|
| Normal | 1 | 6 | 1 | 1 | 3 |
| Repeat | 1 | 6 | 1 | 1 | 3 |
| Block transfer | 1 | 6 | N | N | 3 |

N: Block size (initial setting of CRAH and CRAL)

**Table 7.9 Number of States Required for Each Execution Status**

| Object to be Accessed | | | On-Chip RAM | On-Chip ROM | On-Chip I/O Registers | | External Devices | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Bus width | | | 32 | 16 | 8 | 16 | 8 | | 16 | |
| Access states | | | 1 | 1 | 2 | 2 | 2 | 3 | 2 | 3 |
| Execution status | Vector read | $S_I$ | — | 1 | — | — | 4 | 6+2m | 2 | 3+m |
| | Register information read/write | $S_J$ | 1 | — | — | — | — | — | — | — |
| | Byte data read | $S_K$ | 1 | 1 | 2 | 2 | 2 | 3+m | 2 | 3+m |
| | Word data read | $S_K$ | 1 | 1 | 4 | 2 | 4 | 6+2m | 2 | 3+m |
| | Byte data write | $S_L$ | 1 | 1 | 2 | 2 | 2 | 3+m | 2 | 3+m |
| | Word data write | $S_L$ | 1 | 1 | 4 | 2 | 4 | 6+2m | 2 | 3+m |
| | Internal operation | $S_M$ | 1 | | | | | | | |

The number of execution states is calculated from the formula below. Note that Σ means the sum of all transfers activated by one activation event (the number in which the CHNE bit is set to 1, plus 1).

$$\text{Number of execution states} = I \cdot S_I + \Sigma (J \cdot S_J + K \cdot S_K + L \cdot S_L) + M \cdot S_M$$

For example, when the DTC vector address table is located in on-chip ROM, normal mode is set, and data is transferred from the on-chip ROM to an internal I/O register, the time required for the DTC operation is 13 states. The time from activation to the end of the data write is 10 states.

### 7.3.11 Procedures for Using DTC

**Activation by Interrupt:** The procedure for using the DTC with interrupt activation is as follows:

[1] Set the MRA, MRB, SAR, DAR, CRA, and CRB register information in the on-chip RAM.

[2] Set the start address of the register information in the DTC vector address.

[3] Set the corresponding bit in DTCER to 1.

[4] Set the enable bits for the interrupt sources to be used as the activation sources to 1. The DTC is activated when an interrupt used as an activation source is generated.

[5] After the end of one data transfer, or after the specified number of data transfers have ended, the DTCE bit is cleared to 0 and a CPU interrupt is requested. If the DTC is to continue transferring data, set the DTCE bit to 1.

**Activation by Software:** The procedure for using the DTC with software activation is as follows:

[1] Set the MRA, MRB, SAR, DAR, CRA, and CRB register information in the on-chip RAM.

[2] Set the start address of the register information in the DTC vector address.

[3] Check that the SWDTE bit is 0.

[4] Write 1 to SWDTE bit and the vector number to DTVECR.

[5] Check the vector number written to DTVECR.

[6] After the end of one data transfer, if the DISEL bit is 0 and a CPU interrupt is not requested, the SWDTE bit is cleared to 0. If the DTC is to continue transferring data, set the SWDTE bit to 1. When the DISEL bit is 1, or after the specified number of data transfers have ended, the SWDTE bit is held at 1 and a CPU interrupt is requested.

### 7.3.12 Examples of Use of the DTC

#### (1) Normal Mode

An example is shown in which the DTC is used to receive 128 bytes of data via the SCI.

[1] Set MRA to fixed source address (SM1 = SM0 = 0), incrementing destination address (DM1 = 1, DM0 = 0), normal mode (MD1 = MD0 = 0), and byte size (Sz = 0). The DTS bit can have any value. Set MRB for one data transfer by one interrupt (CHNE = 0, DISEL = 0). Set the SCI RDR address in SAR, the start address of the RAM area where the data will be received in DAR, and 128 (H'0080) in CRA. CRB can be set to any value.

[2] Set the start address of the register information at the DTC vector address.

[3] Set the corresponding bit in DTCER to 1.

[4] Set the SCI to the appropriate receive mode. Set the RIE bit in SCR to 1 to enable the reception complete (RXI) interrupt. Since the generation of a receive error during the SCI reception operation will disable subsequent reception, the CPU should be enabled to accept receive error interrupts.

[5] Each time reception of one byte of data ends on the SCI, the RDRF flag in SSR is set to 1, an RXI interrupt is generated, and the DTC is activated. The receive data is transferred from RDR to RAM by the DTC. DAR is incremented and CRA is decremented. The RDRF flag is automatically cleared to 0.

[6] When CRA becomes 0 after the 128 data transfers have ended, the RDRF flag is held at 1, the DTCE bit is cleared to 0, and an RXI interrupt request is sent to the CPU. The interrupt handling routine should perform wrap-up processing.

**(2) Software Activation**

An example is shown in which the DTC is used to transfer a block of 128 bytes of data by means of software activation. The transfer source address is H'1000 and the destination address is H'2000. The vector number is H'60, so the vector address is H'04C0.

[1] Set MRA to incrementing source address (SM1 = 1, SM0 = 0), incrementing destination address (DM1 = 1, DM0 = 0), block transfer mode (MD1 = 1, MD0 = 0), and byte size (Sz = 0). The DTS bit can have any value. Set MRB for one block transfer by one interrupt (CHNE = 0). Set the transfer source address (H'1000) in SAR, the destination address (H'2000) in DAR, and 128 (H'8080) in CRA. Set 1 (H'0001) in CRB.

[2] Set the start address of the register information at the DTC vector address (H'04C0).

[3] Check that the SWDTE bit in DTVECR is 0. Check that there is currently no transfer activated by software.

[4] Write 1 to the SWDTE bit and the vector number (H'60) to DTVECR. The write data is H'E0.

[5] Read DTVECR again and check that it is set to the vector number (H'60). If it is not, this indicates that the write failed. This is presumably because an interrupt occurred between steps 3 and 4 and led to a different software activation. To activate this transfer, go back to step 3.

[6] If the write was successful, the DTC is activated and a block of 128 bytes of data is transferred.

[7] After the transfer, an SWDTEND interrupt occurs. The interrupt handling routine should clear the SWDTE bit to 0 and perform other wrap-up processing.

## 7.4 Interrupts

An interrupt request is issued to the CPU when the DTC finishes the specified number of data transfers, or a data transfer for which the DISEL bit was set to 1. In the case of interrupt activation, the interrupt set as the activation source is generated. These interrupts to the CPU are subject to CPU mask level and interrupt controller priority level control.

In the case of activation by software, a software activated data transfer end interrupt (SWDTEND) is generated.

When the DISEL bit is 1 and one data transfer has ended, or the specified number of transfers have ended, after data transfer ends, the SWDTE bit is held at 1 and an SWDTEND interrupt is generated. The interrupt handling routine should clear the SWDTE bit to 0.

When the DTC is activated by software, an SWDTEND interrupt is not generated during a data transfer wait or during data transfer even if the SWDTE bit is set to 1.

## 7.5 Usage Notes

**Module Stop:** When the MSTP14 bit in MSTPCR is set to 1, the DTC clock stops, and the DTC enters the module stop state. However, 1 cannot be written in the MSTP14 bit while the DTC is operating.

**On-Chip RAM:** The MRA, MRB, SAR, DAR, CRA, and CRB registers are all located in on-chip RAM. When the DTC is used, the RAME bit in SYSCR must not be cleared to 0.

**DTCE Bit Setting:** For DTCE bit setting, read/write operations must be performed using bit-manipulation instructions such as BSET and BCLR. For the initial setting only, however, when multiple activation sources are set at one time, it is possible to disable interrupts and write after executing a dummy read on the relevant register.

# Section 8 I/O Ports

## 8.1 Overview

The H8S/2345 Series has 10 I/O ports (ports 1, 2, 3, and A to G), and one input-only port (port 4).

Table 8.1 summarizes the port functions. The pins of each port also have other functions.

Each port includes a data direction register (DDR) that controls input/output (not provided for the input-only port), a data register (DR) that stores output data, and a port register (PORT) used to read the pin states.

Ports A to E have a built-in MOS input pull-up function, and in addition to DR and DDR, have a MOS input pull-up control register (PCR) to control the on/off state of MOS input pull-up.

Ports 3 and A include an open-drain control register (ODR) that controls the on/off state of the output buffer PMOS.

Ports 1, and A to F can drive a single TTL load and 90 pF capacitive load, and ports 2, 3, and G can drive a single TTL load and 30 pF capacitive load. All the I/O ports can drive a Darlington transistor when in output mode. Ports 1, and A to C can drive an LED (10 mA sink current).

Port 2, and interrupt input pins ($\overline{\text{IRQ0}}$ to $\overline{\text{IRQ7}}$) are Schmitt-triggered inputs.

For block diagrams of the ports see appendix C, I/O Port Block Diagrams.

**Table 8.1 Port Functions**

| Port | Description | Pins | Mode 1*1 | Mode 2*1, *2 | Mode 3*1, *2 | Mode 4 | Mode 5 | Mode 6*2 | Mode 7*2 |
|---|---|---|---|---|---|---|---|---|---|
| Port 1 | • 8-bit I/O port | $P1_7$/TIOCB2/TCLKD<br>$P1_6$/TIOCA2<br>$P1_5$/TIOCB1/TCLKC<br>$P1_4$/TIOCA1 | 8-bit I/O port also functioning as TPU I/O pins (TCLKA, TCLKB, TCLKC, TCLKD, TIOCA0, TIOCB0, TIOCC0, TIOCD0, TIOCA1, TIOCB1, TIOCA2, TIOCB2) | | | | | | |
| | | $P1_3$/TIOCD0/TCLKB/$A_{23}$<br>$P1_2$/TIOCC0/TCLKA/$A_{22}$<br>$P1_1$/TIOCB0/$A_{21}$<br>$P1_0$/TIOCA0/$A_{20}$ | | | | When DDR = 0: input port also functioning as TPU I/O pins (TCLKA, TCLKB, TIOCA0, TIOCB0, TIOCC0, TIOCD0)<br>When DDR = 1: address output | | | |
| Port 2 | • 8-bit I/O port<br>• Schmitt-triggered input | $P2_7$/TIOCB5/TMO1<br>$P2_6$/TIOCA5/TMO0<br>$P2_5$/TIOCB4/TMCI1<br>$P2_4$/TIOCA4/TMRI1<br>$P2_3$/TIOCD3/TMCI0<br>$P2_2$/TIOCC3/TMRI0<br>$P2_1$/TIOCB3<br>$P2_0$/TIOCA3 | 8-bit I/O port also functioning as TPU I/O pins (TIOCA3, TIOCB3, TIOCC3, TIOCD3, TIOCA4, TIOCB4, TIOCA5, TIOCB5), and 8-bit timer (channels 0 and 1) I/O pins (TMRI0, TMCI0, TMO0, TMRI1, TMCI1, TMO1) | | | | | | |
| Port 3 | • 6-bit I/O port<br>• Open-drain output capability<br>• Schmitt-triggered input ($\overline{IRQ5}$, $\overline{IRQ4}$) | $P3_5$/SCK1/$\overline{IRQ5}$<br>$P3_4$/SCK0/$\overline{IRQ4}$<br>$P3_3$/RxD1<br>$P3_2$/RxD0<br>$P3_1$/TxD1<br>$P3_0$/TxD0 | 6-bit I/O port also functioning as SCI (channels 0 and 1) I/O pins (TxD0, RxD0, SCK0, TxD1, RxD1, SCK1) and interrupt input pins ($\overline{IRQ5}$, $\overline{IRQ4}$) | | | | | | |

**Table 8.1 Port Functions (cont)**

| Port | Description | Pins | Mode 1*1 | Mode 2*1, *2 | Mode 3*1, *2 | Mode 4 | Mode 5 | Mode 6*2 | Mode 7*2 |
|---|---|---|---|---|---|---|---|---|---|
| Port 4 | • 8-bit input port | $P4_7$/AN7/DA1<br>$P4_6$/AN6/DA0<br>$P4_5$/AN5<br>$P4_4$/AN4<br>$P4_3$/AN3<br>$P4_2$/AN2<br>$P4_1$/AN1<br>$P4_0$/AN0 | 8-bit input port also functioning as A/D converter analog inputs (AN7 to AN0) and D/A converter analog outputs (DA1 and DA0) | | | | | | |
| Port A | • 4-bit I/O port<br>• Built-in MOS input pull-up<br>• Open-drain output capability | $PA_3/A_{19}$ to $PA_0/A_{16}$ | I/O ports | | | Address output | | When DDR = 0 (after reset): input ports<br>When DDR = 1: address output | I/O ports |
| Port B | • 8-bit I/O port<br>• Built-in MOS input pull-up | $PB_7/A_{15}$ to $PB_0/A_8$ | Address output | When DDR = 0 (after reset): input port<br>When DDR = 1: address output | I/O port | Address output | | When DDR = 0 (after reset): input port<br>When DDR = 1: address output | I/O port |
| Port C | • 8-bit I/O port<br>• Built-in MOS input pull-up | $PC_7/A_7$ to $PC_0/A_0$ | Address output | When DDR = 0 (after reset): input port<br>When DDR = 1: address output | I/O port | Address output | | When DDR = 0 (after reset): input port<br>When DDR = 1: address output | I/O port |
| Port D | • 8-bit I/O port<br>• Built-in MOS input pull-up | $PD_7/D_{15}$ to $PD_0/D_8$ | Data bus input/output | | I/O port | Data bus input/output | | | I/O port |

**Table 8.1 Port Functions (cont)**

| Port | Description | Pins | Mode 1*1 | Mode 2*1, *2 | Mode 3*1, *2 | Mode 4 | Mode 5 | Mode 6*2 | Mode 7*2 |
|---|---|---|---|---|---|---|---|---|---|
| Port E | • 8-bit I/O port<br>• Built-in MOS input pull-up | $PE_7/D_7$ to $PE_0/D_0$ | In 8-bit bus mode: I/O port<br>In 16-bit bus mode: data bus input/output | | I/O port | In 8-bit bus mode: I/O port<br>In 16-bit bus mode: data bus input/output | | | I/O port |
| Port F | • 8-bit I/O port<br>• Schmitt-triggered input ($\overline{IRQ3}$ to $\overline{IRQ0}$) | $PF_7/ø$ | When DDR = 0: input port<br>When DDR = 1 (after reset): ø output | | When DDR = 0 (after reset): input port<br>When DDR = 1: ø output | When DDR = 0: input port<br>When DDR = 1 (after reset): ø output | | | When DDR = 0 (after reset): input port<br>When DDR = 1: ø output |
| | | $PF_6/\overline{AS}$<br>$PF_5/\overline{RD}$<br>$PF_4/\overline{HWR}$ | $\overline{AS}$, $\overline{RD}$, $\overline{HWR}$, $\overline{LWR}$ output | | I/O port | $\overline{AS}$, $\overline{RD}$, $\overline{HWR}$, $\overline{LWR}$ output | | | I/O port |
| | | $PF_3/\overline{LWR}/\overline{IRQ3}$ | | | I/O port also functioning as interrupt input pins ($\overline{IRQ3}$ to $\overline{IRQ0}$) | | | | I/O port also functioning as interrupt input pins ($\overline{IRQ3}$ to $\overline{IRQ0}$) |
| | | $PF_2/\overline{WAIT}/\overline{IRQ2}$ | When WAITE = 0 (after reset): I/O port also functioning as interrupt input pin ($\overline{IRQ2}$) | | | When WAITE = 0 (after reset): I/O port also functioning as interrupt input pin ($\overline{IRQ2}$) | | | |
| | | | When WAITE = 1: $\overline{WAIT}$ input also functioning as interrupt input pin ($\overline{IRQ2}$) | | | When WAITE = 1: $\overline{WAIT}$ input also functioning as interrupt input pin ($\overline{IRQ2}$) | | | |
| | | $PF_1/\overline{BACK}/\overline{IRQ1}$<br>$PF_0/\overline{BREQ}/\overline{IRQ0}$ | When BRLE = 0 (after reset): I/O port also functioning as interrupt input pins ($\overline{IRQ1}$, $\overline{IRQ0}$) | | | When BRLE = 0 (after reset): I/O port also functioning as interrupt input pins ($\overline{IRQ1}$, $\overline{IRQ0}$) | | | |
| | | | When BRLE = 1: $\overline{BREQ}$ input, $\overline{BACK}$ output also functioning as interrupt input pins ($\overline{IRQ1}$, $\overline{IRQ0}$) | | | When BRLE = 1: $\overline{BREQ}$ input, $\overline{BACK}$ output also functioning as interrupt input pins ($\overline{IRQ1}$, $\overline{IRQ0}$) | | | |

**Table 8.1 Port Functions (cont)**

| Port | Description | Pins | Mode 1*1 | Mode 2*1, *2 | Mode 3*1, *2 | Mode 4 | Mode 5 | Mode 6*2 | Mode 7*2 |
|---|---|---|---|---|---|---|---|---|---|
| Port G | • 5-bit I/O port<br>• Schmitt-triggered input ($\overline{IRQ7}$, $\overline{IRQ6}$) | $PG_4/\overline{CS0}$ | When DDR = 0*3: input port<br>When DDR = 1*4: $\overline{CS0}$ output | | I/O port also functioning as interrupt input pins ($\overline{IRQ7}$, $\overline{IRQ6}$) | When DDR = 0*3: input port<br>When DDR = 1*4: $\overline{CS0}$ output | | | I/O port also functions as interrupt input pins ($\overline{IRQ7}$, $\overline{IRQ6}$) and A/D converter input pin ($\overline{ADTRG}$) |
| | | $PG_3/\overline{CS1}$<br>$PG_2/\overline{CS2}$<br>$PG_1/\overline{CS3}/\overline{IRQ7}$ | I/O port also functioning as interrupt input pins ($\overline{IRQ7}$, $\overline{IRQ6}$) and A/D converter input pin ($\overline{ADTRG}$) | | and A/D converter input pin ($\overline{ADTRG}$) | When DDR = 0 (after reset): input port also functioning as interrupt input pin ($\overline{IRQ7}$)<br>When DDR = 1: $\overline{CS1}$, $\overline{CS2}$, $\overline{CS3}$ output also functioning as interrupt input pin ($\overline{IRQ7}$) | | | |
| | | $PG_0/\overline{IRQ6}/\overline{ADTRG}$ | | | | I/O port also functioning as interrupt input pin ($\overline{IRQ6}$) and A/D converter input pin ($\overline{ADTRG}$) | | | |

Notes: 1. Modes 1 to 3 are not available on the F-ZTAT version.
2. Modes 2, 3, 6, and 7 are not available on the ROMless version.
3. After a reset in mode 2, 6, 10 or 14
4. After a reset in mode 1, 4 or 5

## 8.2 Port 1

### 8.2.1 Overview

Port 1 is an 8-bit I/O port. Port 1 pins also function as TPU I/O pins (TCLKA, TCLKB, TCLKC, TCLKD, TIOCA0, TIOCB0, TIOCC0, TIOCD0, TIOCA1, TIOCB1, TIOCA2, and TIOCB2) and an address bus output function. Port 1 pin functions change according to the operating mode.

Figure 8.1 shows the port 1 pin configuration.

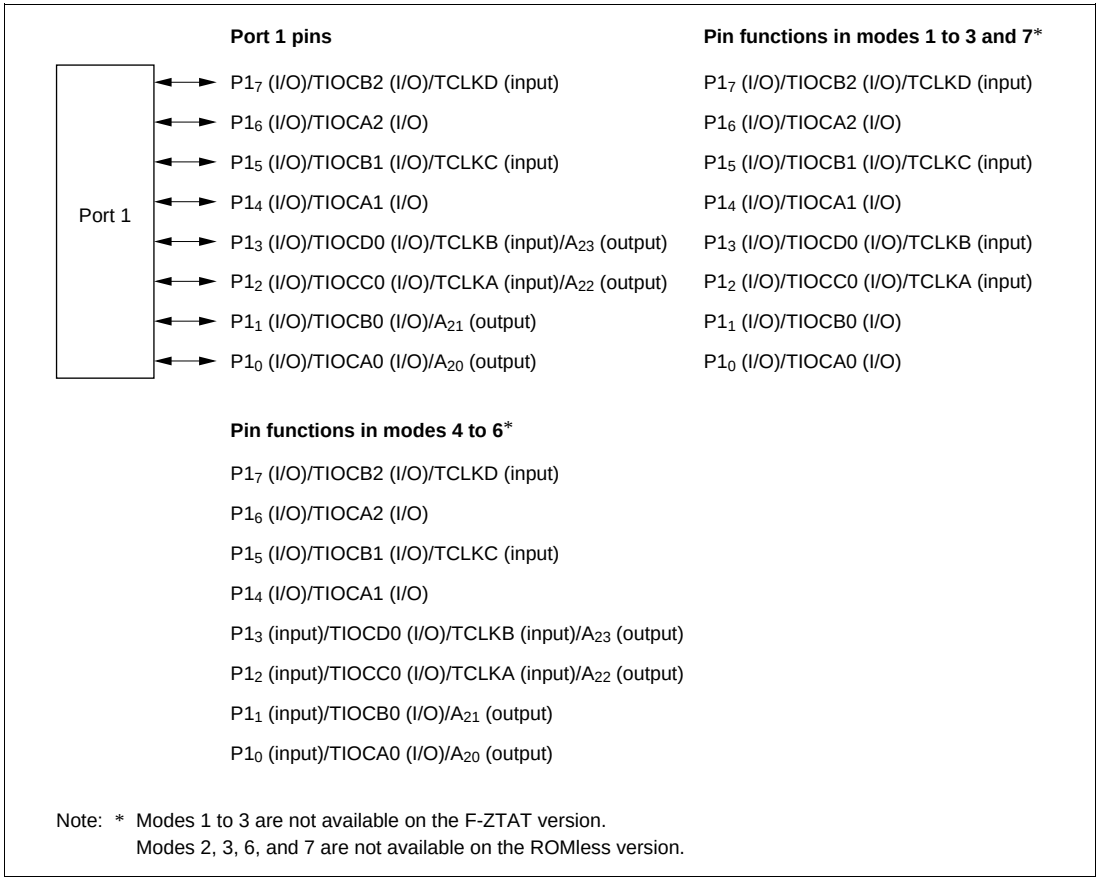



**Figure 8.1 Port 1 Pin Functions**

### 8.2.2 Register Configuration

Table 8.2 shows the port 1 register configuration.

**Table 8.2 Port 1 Registers**

| Name | Abbreviation | R/W | Initial Value | Address* |
|---|---|---|---|---|
| Port 1 data direction register | P1DDR | W | H'00 | H'FEB0 |
| Port 1 data register | P1DR | R/W | H'00 | H'FF60 |
| Port 1 register | PORT1 | R | Undefined | H'FF50 |

Note: * Lower 16 bits of the address.

**Port 1 Data Direction Register (P1DDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | P17DDR | P16DDR | P15DDR | P14DDR | P13DDR | P12DDR | P11DDR | P10DDR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | W | W | W | W | W | W | W | W |

P1DDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port 1. P1DDR cannot be read; if it is, an undefined value will be read.

Setting a P1DDR bit to 1 makes the corresponding port 1 pin an output pin, while clearing the bit to 0 makes the pin an input pin.

P1DDR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode. As the TPU is initialized by a manual reset, the pin states are determined by the P1DDR and P1DR specifications.

Whether the address output pins maintain their output state or go to the high-impedance state in a transition to software standby mode is selected by the OPE bit in SBYCR.

- Modes 1 to 3 and 7*

  The corresponding port 1 pins are output ports when P1DDR is set to 1, and input ports when cleared to 0.

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

- Modes 4 to 6*

  The corresponding port 1 pins are address outputs when P13DDR to P10DDR are set to 1, and input ports when cleared to 0.

  The corresponding port 1 pins are output ports when P17DDR to P14DDR are set to 1, and input ports when cleared to 0.

**Port 1 Data Register (P1DR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | P17DR | P16DR | P15DR | P14DR | P13DR | P12DR | P11DR | P10DR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

P1DR is an 8-bit readable/writable register that stores output data for the port 1 pins ($P1_7$ to $P1_0$).

P1DR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port 1 Register (PORT1)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 |
| Initial value | : | —* | —* | —* | —* | —* | —* | —* | —* |
| R/W | : | R | R | R | R | R | R | R | R |

Note: * Determined by state of pins $P1_7$ to $P1_0$.

PORT1 is an 8-bit read-only register that shows the pin states. It cannot be written to. Writing of output data for the port 1 pins ($P1_7$ to $P1_0$) must always be performed on P1DR.

If a port 1 read is performed while P1DDR bits are set to 1, the P1DR values are read. If a port 1 read is performed while P1DDR bits are cleared to 0, the pin states are read.

After a power-on reset and in hardware standby mode, PORT1 contents are determined by the pin states, as P1DDR and P1DR are initialized. PORT1 retains its prior state after a manual reset, and in software standby mode.

### 8.2.3 Pin Functions

Port 1 pins also function as TPU I/O pins (TCLKA, TCLKB, TCLKC, TCLKD, TIOCA0, TIOCB0, TIOCC0, TIOCD0, TIOCA1, TIOCB1, TIOCA2, and TIOCB2) and address output pins ($A_{23}$ to $A_{20}$). Port 1 pin functions are shown in table 8.3.

**Table 8.3 Port 1 Pin Functions**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P1_7$/TIOCB2/ TCLKD | The pin function is switched as shown below according to the combination of the TPU channel 2 setting by bits MD3 to MD0 in TMDR2, bits IOB3 to IOB0 in TIOR2, bits CCLR1 and CCLR0 in TCR2, bits TPSC2 to TPSC0 in TCR0 and TCR5, and bit P17DDR. |

| TPU Channel 2 Setting | Table Below (1) | Table Below (2) | |
|---|---|---|---|
| P17DDR | — | 0 | 1 |
| Pin function | TIOCB2 output | $P1_7$ input | $P1_7$ output |
| | | TIOCB2 input *[1] | |
| | TCLKD input *[2] | | |

| TPU Channel 2 Setting | (2) | (1) | (2) | (2) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000, B'01xx | | B'0010 | B'0011 | | |
| IOB3 to IOB0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | — | B'xx00 | Other than B'xx00 | |
| CCLR1, CCLR0 | — | — | — | — | Other than B'10 | B'10 |
| Output function | — | Output compare output | — | — | PWM mode 2 output | — |

x: Don't care

Notes: 1. TIOCB2 input when TPU channel 2 is in normal operation mode (MD3 to MD0 = B'0000) and input capture is set (IOB3 to IOB0 = B'1xxx).

2. TCLKD input when the setting for either TCR0 or TCR5 is: TPSC2 to TPSC0 = B'111.
TCLKD input when channels 2 and 4 are set to phase counting mode (MD3 to MD0 = B'01xx).

**Table 8.3 Port 1 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P1_6$/TIOCA2 | The pin function is switched as shown below according to the combination of the TPU channel 2 setting by bits MD3 to MD0 in TMDR2, bits IOA3 to IOA0 in TIOR2, bits CCLR1 and CCLR0 in TCR2, and bit P16DDR. |

| TPU Channel 2 Setting | Table Below (1) | Table Below (2) | |
|---|---|---|---|
| P16DDR | — | 0 | 1 |
| Pin function | TIOCA2 output | $P1_6$ input | $P1_6$ output |
| | | TIOCA2 input *1 | |

| TPU Channel 2 Setting | (2) | (1) | (2) | (1) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000, B'01xx | | B'001x | B'0011 | B'0011 | |
| IOA3 to IOA0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | B'xx00 | Other than B'xx00 | | |
| CCLR1, CCLR0 | — | — | — | — | Other than B'01 | B'01 |
| Output function | — | Output compare output | — | PWM mode 1 output *2 | PWM mode 2 output | — |

x: Don't care

Notes: 1. TIOCA2 input when TPU channel 2 is in normal operation mode (MD3 to MD0 = B'0000) and input capture is set (IOA3 to IOA0 = B'10xx).
2. TIOCB2 output is disabled.

**Table 8.3 Port 1 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P1_5$/TIOCB1/ TCLKC | The pin function is switched as shown below according to the combination of the TPU channel 1 setting by bits MD3 to MD0 in TMDR1, bits IOB3 to IOB0 in TIOR1, bits CCLR1 and CCLR0 in TCR1, bits TPSC2 to TPSC0 in TCR0, TCR2, TCR4, and TCR5, and bit P15DDR. |

| TPU Channel 1 Setting | Table Below (1) | Table Below (2) | |
|---|---|---|---|
| P15DDR | — | 0 | 1 |
| Pin function | TIOCB1 output | $P1_5$ input | $P1_5$ output |
| | | TIOCB1 input *[1] | |
| | TCLKC input *[2] | | |

| TPU Channel 1 Setting | (2) | (1) | (2) | (2) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000, B'01xx | | B'0010 | B'0011 | | |
| IOB3 to IOB0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | — | B'xx00 | Other than B'xx00 | |
| CCLR1, CCLR0 | — | — | — | — | Other than B'10 | B'10 |
| Output function | — | Output compare output | — | — | PWM mode 2 output | — |

x: Don't care

Notes: 1. TIOCB1 input when TPU channel 1 is in normal operation mode (MD3 to MD0 = B'0000) and input capture is set (IOB3 to IOB0 = B'10xx).

2. TCLKC input when the setting for either TCR0 or TCR2 is: TPSC2 to TPSC0 = B'110; or when the setting for either TCR4 or TCR5 is TPSC2 to TPSC0 = B'101.
   TCLKC input when channels 2 and 4 are set to phase counting mode (MD3 to MD0 = B'01xx).

**Table 8.3 Port 1 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P1_4$/TIOCA1 | The pin function is switched as shown below according to the combination of the TPU channel 1 setting by bits MD3 to MD0 in TMDR1, bits IOA3 to IOA0 in TIOR1, bits CCLR1 and CCLR0 in TCR1, and bit P14DDR. |

| TPU Channel 1 Setting | Table Below (1) | Table Below (2) | |
|---|---|---|---|
| P14DDR | — | 0 | 1 |
| Pin function | TIOCA1 output | $P1_4$ input | $P1_4$ output |
| | | TIOCA1 input *[1] | |

| TPU Channel 1 Setting | (2) | (1) | (2) | (1) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000, B'01xx | | B'001x | B'0010 | B'0011 | |
| IOA3 to IOA0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | B'xx00 | Other than B'xx00 | | |
| CCLR1, CCLR0 | — | — | — | — | Other than B'01 | B'01 |
| Output function | — | Output compare output | — | PWM mode 1 output*[2] | PWM mode 2 output | — |

x: Don't care

Notes: 1. TIOCA1 input when TPU channel 1 is in normal operation mode (MD3 to MD0 = B'0000) and input capture is set (IOA3 to IOA0 = B'10xx).

2. TIOCB1 output is disabled.

**Table 8.3 Port 1 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P1_3$/TIOCD0/ TCLKB/$A_{23}$ | The pin function is switched as shown below according to the combination of the operating mode, TPU channel 0 setting (by bits MD3 to MD0 in TMDR0, bits IOD3 to IOD0 in TIOR0L, and bits CCLR2 to CCLR0 in TCR0), bits TPSC2 to TPSC0 in TCR0 to TCR2, and bit P13DDR. |

| Operating Mode | Modes 1, 2, 3, 7*[1] | | | Modes 4, 5, 6*[1] | | | |
|---|---|---|---|---|---|---|---|
| TPU Channel 0 Setting | Table Below (1) | Table Below (2) | | Table Below (1) | | Table Below (2) | |
| P13DDR | — | 0 | 1 | 0 | 1 | 0 | 1 |
| Pin function | TIOCD0 output | $P1_3$ input | $P1_3$ output | TIOCD0 output | $A_{23}$ output | $P1_3$ input | $A_{23}$ output |
| | | TIOCD0 input*[2] | | | | TIOCD0 input*[2] | |
| | TCLKB input*[3] | | | | | | |

| TPU Channel 0 Setting | (2) | (1) | (2) | (2) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000 | | B'0010 | B'0011 | | |
| IOD3 to IOD0 | B'0000 B'0100 B'1xxx | B'0001 to B'0011 B'0101 to B'0111 | — | B'xx00 | Other than B'xx00 | |
| CCLR2 to CCLR0 | — | — | — | — | Other than B'110 | B'110 |
| Output function | — | Output compare output | — | — | PWM mode 2 output | — |

x: Don't care

Notes: 1. Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

2. TIOCD0 input when TPU channel 0 is in normal operation mode (MD3 to MD0 = B'0000) and input capture is set (IOD3 to IOD0 = B'10xx).

3. TCLKB input when the TCR0, TCR1, or TCR2 setting is: TPSC2 to TPSC0 = B'101.

   TCLKB input when channels 1 and 5 are set to phase counting mode (MD3 to MD0 = B'01xx).

**Table 8.3 Port 1 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P1_2$/TIOCC0/ TCLKA/$A_{22}$ | The pin function is switched as shown below according to the combination of the operating mode, TPU channel 0 setting (by bits MD3 to MD0 in TMDR0, bits IOC3 to IOC0 in TIOR0L, and bits CCLR2 to CCLR0 in TCR0), bits TPSC2 to TPSC0 in TCR0 to TCR2, and bit P12DDR. |

| Operating Mode | Modes 1, 2, 3, 7*[1] | | | Modes 4, 5, 6*[1] | | | |
|---|---|---|---|---|---|---|---|
| TPU Channel 0 Setting | Table Below (1) | Table Below (2) | | Table Below (1) | | Table Below (2) | |
| P12DDR | — | 0 | 1 | 0 | 1 | 0 | 1 |
| Pin function | TIOCC0 output | $P1_2$ input | $P1_2$ output | TIOCC0 output | $A_{22}$ output | $P1_2$ input | $A_{22}$ output |
| | | TIOCC0 input*[2] | | | | TIOCC0 input*[2] | |
| | TCLKA input*[3] | | | | | | |

| TPU Channel 0 Setting | (2) | (1) | (2) | (1) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000 | | B'001x | B'0010 | B'0011 | |
| IOC3 to IOC0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | B'xx00 | Other than B'xx00 | | |
| CCLR2 to CCLR0 | — | — | — | — | Other than B'101 | B'101 |
| Output function | — | Output compare output | — | PWM mode 1 output*[4] | PWM mode 2 output | — |

x: Don't care

Notes: 1. Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

2. TIOCC0 input when TPU channel 0 is in normal operation mode (MD3 to MD0 = B'0000) and input capture is set (IOC3 to IOC0 = B'10xx).

3. TCLKA input when the TCR0 to TCR5 setting is: TPSC2 to TPSC0 = B'100.
TCLKA input when channel 1 and 5 are set to phase counting mode (MD3 to MD0 = B'01xx).

4. TIOCD0 output is disabled.
When BFA = 1 or BFB = 1 in TMDR0, output is disabled and setting (2) applies.

**Table 8.3 Port 1 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P1_1$/TIOCB0/ $A_{21}$ | The pin function is switched as shown below according to the combination of the operating mode, TPU channel 0 setting (by bits MD3 to MD0 in TMDR0, bits IOB3 to IOB0 in TIOR0H, and bits CCLR2 to CCLR0 in TCR0), and bit P11DDR. |

| Operating Mode | Modes 1, 2, 3, 7*[1] | | | Modes 4, 5, 6*[1] | | | |
|---|---|---|---|---|---|---|---|
| TPU Channel 0 Setting | Table Below (1) | Table Below (2) | | Table Below (1) | | Table Below (2) | |
| P11DDR | — | 0 | 1 | 0 | 1 | 0 | 1 |
| Pin function | TIOCB0 output | $P1_1$ input | $P1_1$ output | TIOCB0 output | $A_{21}$ output | $P1_1$ input | $A_{21}$ output |
| | | TIOCB0 input*[2] | | | | TIOCB0 input*[2] | |

| TPU Channel 0 Setting | (2) | (1) | (2) | (2) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000 | | B'0010 | B'0011 | | |
| IOB3 to IOB0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | — | B'xx00 | Other than B'xx00 | |
| CCLR2 to CCLR0 | — | — | — | — | Other than B'010 | B'010 |
| Output function | — | Output compare output | — | — | PWM mode 2 output | — |

x: Don't care

Notes: 1. Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.
2. TIOCB0 input when TPU channel 0 is in normal operation mode (MD3 to MD0 = B'0000) and input capture is set (IOB3 to IOB0 = B'10xx).

**Table 8.3 Port 1 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P1_0$/TIOCA0/ $A_{20}$ | The pin function is switched as shown below according to the combination of the operating mode, TPU channel 0 setting (by bits MD3 to MD0 in TMDR0, bits IOA3 to IOA0 in TIOR0H, and bits CCLR2 to CCLR0 in TCR0), and bit P10DDR. |

| Operating Mode | Modes 1, 2, 3, 7*[1] | | | Modes 4, 5, 6*[1] | | | |
|---|---|---|---|---|---|---|---|
| TPU Channel 0 Setting | Table Below (1) | Table Below (2) | | Table Below (1) | | Table Below (2) | |
| P10DDR | — | 0 | 1 | 0 | 1 | 0 | 1 |
| Pin function | TIOCA0 output | $P1_0$ input | $P1_0$ output | TIOCA0 output | $A_{20}$ output | $P1_0$ input | $A_{20}$ output |
| | | TIOCA0 input*[2] | | | | TIOCA0 input*[2] | |

| TPU Channel 0 Setting | (2) | (1) | (2) | (1) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000 | | B'001x | B'0010 | B'0011 | |
| IOA3 to IOA0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | B'xx00 | Other than B'xx00 | | |
| CCLR2 to CCLR0 | — | — | — | — | Other than B'001 | B'001 |
| Output function | — | Output compare output | — | PWM mode 1 output*[3] | PWM mode 2 output | — |

x: Don't care

Notes: 1. Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.
2. TIOCA0 input when TPU channel 0 is in normal operation mode (MD3 to MD0 = B'0000) and input capture is set (IOA3 to IOA0 = B'10xx).
3. TIOCB0 output is disabled.

## 8.3 Port 2

### 8.3.1 Overview

Port 2 is an 8-bit I/O port. Port 2 pins also function as TPU I/O pins (TIOCA3, TIOCB3, TIOCC3, TIOCD3, TIOCA4, TIOCB4, TIOCA5, and TIOCB5), and 8-bit timer I/O pins (TMRI0, TMCI0, TMO0, TMRI1, TMCI1, and TMO1). Port 2 pin functions are the same in all operating modes. Port 2 uses Schmitt-triggered input.

Figure 8.2 shows the port 2 pin configuration.

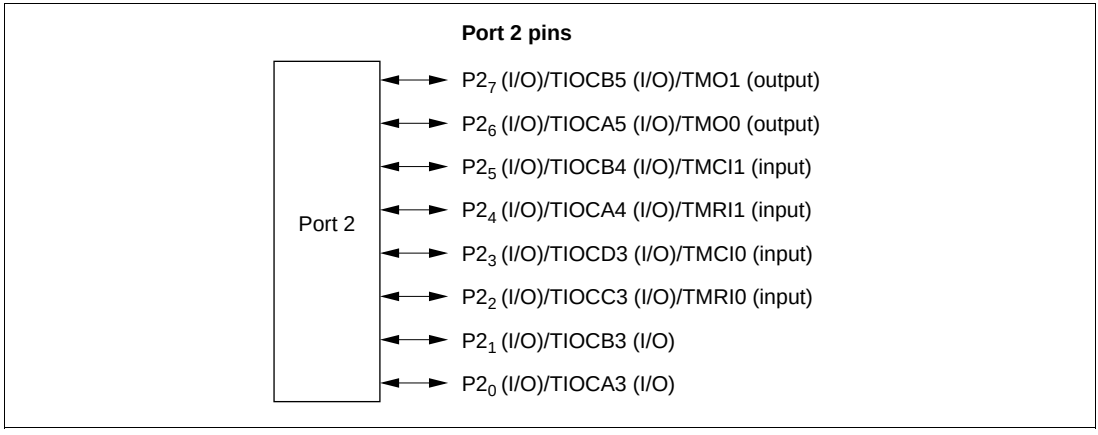



**Figure 8.2 Port 2 Pin Functions**

### 8.3.2 Register Configuration

Table 8.4 shows the port 2 register configuration.

**Table 8.4 Port 2 Registers**

| Name | Abbreviation | R/W | Initial Value | Address* |
|---|---|---|---|---|
| Port 2 data direction register | P2DDR | W | H'00 | H'FEB1 |
| Port 2 data register | P2DR | R/W | H'00 | H'FF61 |
| Port 2 register | PORT2 | R | Undefined | H'FF51 |

Note: * Lower 16 bits of the address.

**Port 2 Data Direction Register (P2DDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | P27DDR | P26DDR | P25DDR | P24DDR | P23DDR | P22DDR | P21DDR | P20DDR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | W | W | W | W | W | W | W | W |

P2DDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port 2. P2DDR cannot be read; if it is, an undefined value will be read.

Setting a P2DDR bit to 1 makes the corresponding port 2 pin an output pin, while clearing the bit to 0 makes the pin an input pin.

P2DDR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode. As the TPU and 8-bit timer are initialized by a manual reset, the pin states are determined by the P2DDR and P2DR specifications.

**Port 2 Data Register (P2DR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | P27DR | P26DR | P25DR | P24DR | P23DR | P22DR | P21DR | P20DR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

P2DR is an 8-bit readable/writable register that stores output data for the port 2 pins ($P2_7$ to $P2_0$).

P2DR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port 2 Register (PORT2)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | P27 | P26 | P25 | P24 | P23 | P22 | P21 | P20 |
| Initial value | : | —* | —* | —* | —* | —* | —* | —* | —* |
| R/W | : | R | R | R | R | R | R | R | R |

Note: * Determined by state of pins $P2_7$ to $P2_0$.

PORT2 is an 8-bit read-only register that shows the pin states. It cannot be written to. Writing of output data for the port 2 pins ($P2_7$ to $P2_0$) must always be performed on P2DR.

If a port 2 read is performed while P2DDR bits are set to 1, the P2DR values are read. If a port 2 read is performed while P2DDR bits are cleared to 0, the pin states are read.

After a power-on reset and in hardware standby mode, PORT2 contents are determined by the pin states, as P2DDR and P2DR are initialized. PORT2 retains its prior state after a manual reset, and in software standby mode.

### 8.3.3 Pin Functions

Port 2 pins also function as TPU I/O pins (TIOCA3, TIOCB3, TIOCC3, TIOCD3, TIOCA4, TIOCB4, TIOCA5, and TIOCB5), and 8-bit timer I/O pins (TMRI0, TMCI0, TMO0, TMRI1, TMCI1, and TMO1). Port 2 pin functions are shown in table 8.5.

**Table 8.5 Port 2 Pin Functions**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P2_7$/TIOCB5/ TMO1 | The pin function is switched as shown below according to the combination of the TPU channel 5 setting by bits MD3 to MD0 in TMDR5, bits IOB3 to IOB0 in TIOR5, bits CCLR1 and CCLR0 in TCR5, bits OS3 to OS0 in TCSR1, and bit P27DDR. |

| OS3 to OS0 | All 0 | | | Any 1 |
|---|---|---|---|---|
| TPU Channel 5 Setting | Table Below (1) | Table Below (2) | | — |
| P27DDR | — | 0 | 1 | — |
| Pin function | TIOCB5 output | $P2_7$ input | $P2_7$ output | TMO1 output |
| | | TIOCB5 input * | | |

| TPU Channel 5 Setting | (2) | (1) | (2) | (2) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000, B'01xx | | B'0010 | B'0011 | | |
| IOB3 to IOB0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | — | B'xx00 | Other than B'xx00 | |
| CCLR1, CCLR0 | — | — | — | — | Other than B'10 | B'10 |
| Output function | — | Output compare output | — | — | PWM mode 2 output | — |

x: Don't care

Note: * TIOCB5 input when TPU channel 5 is in normal operation mode (MD3 to MD0 = B'0000) and input capture is set (IOB3 to IOB0 = B'1xxx).

**Table 8.5 Port 2 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P2_6$/TIOCA5/ TMO0 | The pin function is switched as shown below according to the combination of the TPU channel 5 setting by bits MD3 to MD0 in TMDR5, bits IOA3 to IOA0 in TIOR5, bits CCLR1 and CCLR0 in TCR5, bits OS3 to OS0 in TCSR0, and bit P26DDR. |

| OS3 to OS0 | All 0 | | | Any 1 |
|---|---|---|---|---|
| TPU Channel 5 Setting | Table Below (1) | Table Below (2) | | — |
| P26DDR | — | 0 | 1 | — |
| NDER6 | — | — | 0 | — |
| Pin function | TIOCA5 output | $P2_6$ input | $P2_6$ output | TMO0 output |
| | | TIOCA5 input *[1] | | |

| TPU Channel 5 Setting | (2) | (1) | (2) | (1) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000, B'01xx | | B'001x | B'0010 | B'0011 | |
| IOA3 to IOA0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | B'xx00 | Other than B'xx00 | | |
| CCLR1, CCLR0 | — | — | — | — | Other than B'01 | B'01 |
| Output function | — | Output compare output | — | PWM mode 1 output*[2] | PWM mode 2 output | — |

x: Don't care

Notes: 1. TIOCA5 input when TPU channel 5 is in normal operation mode (MD3 to MD0 = B'0000) and input capture is set (IOA3 to IOA0 = B'1xxx).

2. TIOCB5 output is disabled.

**Table 8.5 Port 2 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P2_5$/TIOCB4/ TMCI1 | This pin is used as the 8-bit timer external clock input pin when external clock is selected with bits CKS2 to CKS0 in TCR1.<br>The pin function is switched as shown below according to the combination of the TPU channel 4 setting by bits MD3 to MD0 in TMDR4 and bits IOB3 to IOB0 in TIOR4, bits CCLR1 and CCLR0 in TCR4, and bit P25DDR. |

| TPU Channel 4 Setting | Table Below (1) | Table Below (2) | |
|---|---|---|---|
| P25DDR | — | 0 | 1 |
| Pin function | TIOCB4 output | $P2_5$ input | $P2_5$ output |
| | | TIOCB4 input * | |
| | TMCI1 input | | |

| TPU Channel 4 Setting | (2) | (1) | (2) | (2) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000, B'01xx | | B'0010 | B'0011 | | |
| IOB3 to IOB0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | — | B'xx00 | Other than B'xx00 | |
| CCLR1, CCLR0 | — | — | — | — | Other than B'10 | B'10 |
| Output function | — | Output compare output | — | — | PWM mode 2 output | — |

x: Don't care

Note: * TIOCB4 input when TPU channel 4 is in normal operation mode (MD3 to MD0 = B'0000) and input capture is set (IOB3 to IOB0 = B'10xx).

**Table 8.5 Port 2 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P2_4$/TIOCA4/ TMRI1 | This pin is used as the 8-bit timer counter reset pin when bits CCLR1 and CCLR0 in TCR1 are both set to 1.<br>The pin function is switched as shown below according to the combination of the TPU channel 4 setting by bits MD3 to MD0 in TMDR4, bits IOA3 to IOA0 in TIOR4, bits CCLR1 and CCLR0 in TCR4, and bit P24DDR. |

| TPU Channel 4 Setting | Table Below (1) | Table Below (2) | |
|---|---|---|---|
| P24DDR | — | 0 | 1 |
| Pin function | TIOCA4 output | $P2_4$ input | $P2_4$ output |
| | | TIOCA4 input *1 | |
| | TMRI1 input | | |

| TPU Channel 4 Setting | (2) | (1) | (2) | (1) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000, B'01xx | | B'001x | B'0010 | B'0011 | |
| IOA3 to IOA0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | B'xx00 | Other than B'xx00 | | |
| CCLR1, CCLR0 | — | — | — | — | Other than B'01 | B'01 |
| Output function | — | Output compare output | — | PWM mode 1 output*2 | PWM mode 2 output | — |

x: Don't care

Notes: 1. TIOCA4 input when TPU channel 4 is in normal operation mode (MD3 to MD0 = B'0000) and input capture is set (IOA3 to IOA0 = B'10xx).

2. TIOCB4 output is disabled.

**Table 8.5 Port 2 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P2_3$/TIOCD3/ TMCI0 | This pin is used as the 8-bit timer external clock input pin when external clock is selected with bits CKS2 to CKS0 in TCR0.<br>The pin function is switched as shown below according to the combination of the TPU channel 3 setting by bits MD3 to MD0 in TMDR3, bits IOD3 to IOD0 in TIOR3L, bits CCLR2 to CCLR0 in TCR3, and bit P23DDR. |

| TPU Channel 3 Setting | Table Below (1) | Table Below (2) | |
|---|---|---|---|
| P23DDR | — | 0 | 1 |
| Pin function | TIOCD3 output | $P2_3$ input | $P2_3$ output |
| | | TIOCD3 input * | |
| | TMCI0 input | | |

| TPU Channel 3 Setting | (2) | (1) | (2) | (2) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000 | | B'0010 | B'0011 | | |
| IOD3 to IOD0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | — | B'xx00 | Other than B'xx00 | |
| CCLR2 to CCLR0 | — | — | — | — | Other than B'110 | B'110 |
| Output function | — | Output compare output | — | — | PWM mode 2 output | — |

x: Don't care

Note: * TIOCD3 input when TPU channel 3 is in normal operation mode (MD3 to MD0 = B'0000) and input capture is set (IOD3 to IOD0 = B'10xx).

**Table 8.5 Port 2 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P2_2$/TIOCC3/ TMCI0 | This pin is used as the 8-bit timer counter reset pin when bits CCLR1 and CCLR0 in TCR0 are both set to 1.<br>The pin function is switched as shown below according to the combination of the TPU channel 3 setting by bits MD3 to MD0 in TMDR3, bits IOC3 to IOC0 in TIOR3L, bits CCLR2 to CCLR0 in TCR3, and bit P22DDR. |

| TPU Channel 3 Setting | Table Below (1) | Table Below (2) | |
|---|---|---|---|
| P22DDR | — | 0 | 1 |
| Pin function | TIOCC3 output | $P2_2$ input | $P2_2$ output |
| | | TIOCC3 input *[1] | |
| | TMRI0 input | | |

| TPU Channel 3 Setting | (2) | (1) | (2) | (1) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000 | | B'001x | B'0010 | B'0011 | |
| IOC3 to IOC0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | B'xx00 | Other than B'xx00 | | |
| CCLR2 to CCLR0 | — | — | — | — | Other than B'101 | B'101 |
| Output function | — | Output compare output | — | PWM mode 1 output*[2] | PWM mode 2 output | — |

x: Don't care

Notes: 1. TIOCC3 input when TPU channel 3 is in normal operation mode (MD3 to MD0 = B'0000) and input capture is set (IOC3 to IOC0 = B'10xx).

2. TIOCD3 output is disabled.
When BFA = 1 or BFB = 1 in TMDR3, output is disabled and setting (2) applies.

**Table 8.5 Port 2 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P2_1$/TIOCB3 | The pin function is switched as shown below according to the combination of the TPU channel 3 setting by bits MD3 to MD0 in TMDR3, bits IOB3 to IOB0 in TIOR3H, bits CCLR2 to CCLR0 in TCR3, and bit P21DDR. |

| TPU Channel 3 Setting | Table Below (1) | Table Below (2) | |
|---|---|---|---|
| P21DDR | — | 0 | 1 |
| Pin function | TIOCB3 output | $P2_1$ input | $P2_1$ output |
| | | TIOCB3 input * | |

| TPU Channel 3 Setting | (2) | (1) | (2) | (2) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000 | | B'0010 | B'0011 | | |
| IOB3 to IOB0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | — | B'xx00 | Other than B'xx00 | |
| CCLR2 to CCLR0 | — | — | — | — | Other than B'010 | B'010 |
| Output function | — | Output compare output | — | — | PWM mode 2 output | — |

x: Don't care

Note: * TIOCB3 input when TPU channel 3 is in normal operation mode (MD3 to MD0 = B'0000) and input capture is set (IOB3 to IOB0 = B'10xx).

**Table 8.5 Port 2 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P2_0$/TIOCA3 | The pin function is switched as shown below according to the combination of the TPU channel 3 setting by bits MD3 to MD0 in TMDR3, bits IOA3 to IOA0 in TIOR3H, bits CCLR2 to CCLR0 in TCR3, and bit P20DDR. |

| TPU Channel 3 Setting | Table Below (1) | Table Below (2) | |
|---|---|---|---|
| P20DDR | — | 0 | 1 |
| Pin function | TIOCA3 output | $P2_0$ input | $P2_0$ output |
| | | TIOCA3 input *[1] | |

| TPU Channel 3 Setting | (2) | (1) | (2) | (1) | (1) | (2) |
|---|---|---|---|---|---|---|
| MD3 to MD0 | B'0000 | | B'001x | B'0010 | B'0011 | |
| IOA3 to IOA0 | B'0000<br>B'0100<br>B'1xxx | B'0001 to B'0011<br>B'0101 to B'0111 | B'xx00 | Other than B'xx00 | | |
| CCLR2 to CCLR0 | — | — | — | — | Other than B'001 | B'001 |
| Output function | — | Output compare output | — | PWM mode 1 output*[2] | PWM mode 2 output | — |

x: Don't care

Notes: 1. TIOCA3 input when TPU channel 3 is in normal operation mode (MD3 to MD0 = B'0000) and input capture is set (IOA3 to IOA0 = B'10xx).

2. TIOCB3 output is disabled.

## 8.4 Port 3

### 8.4.1 Overview

Port 3 is a 6-bit I/O port. Port 3 pins also function as SCI I/O pins (TxD0, RxD0, SCK0, TxD1, RxD1, and SCK1) and interrupt input pins ($\overline{IRQ4}$, $\overline{IRQ5}$). Port 3 pin functions are the same in all operating modes. The interrupt input pins ($\overline{IRQ4}$, $\overline{IRQ5}$) are Schmitt-triggered inputs.

Figure 8.3 shows the port 3 pin configuration.

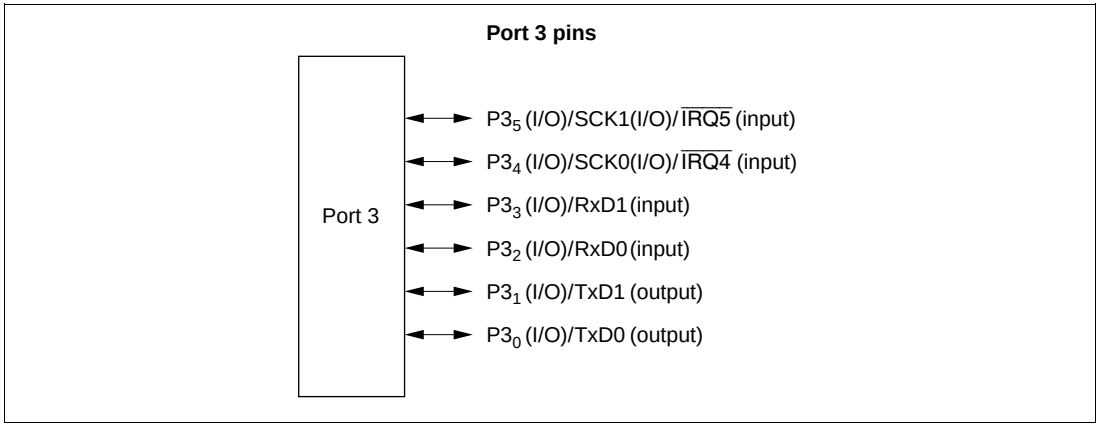



**Figure 8.3 Port 3 Pin Functions**

### 8.4.2 Register Configuration

Table 8.6 shows the port 3 register configuration.

**Table 8.6 Port 3 Registers**

| Name | Abbreviation | R/W | Initial Value*[1] | Address*[2] |
|---|---|---|---|---|
| Port 3 data direction register | P3DDR | W | H'00 | H'FEB2 |
| Port 3 data register | P3DR | R/W | H'00 | H'FF62 |
| Port 3 register | PORT3 | R | Undefined | H'FF52 |
| Port 3 open drain control register | P3ODR | R/W | H'00 | H'FF76 |

Notes: 1. Value of bits 5 to 0.
2. Lower 16 bits of the address.

**Port 3 Data Direction Register (P3DDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | P35DDR | P34DDR | P33DDR | P32DDR | P31DDR | P30DDR |
| Initial value | : | Undefined | Undefined | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | W | W | W | W | W | W |

P3DDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port 3. Bits 7 and 6 are reserved. P3DDR cannot be read; if it is, an undefined value will be read.

Setting a P3DDR bit to 1 makes the corresponding port 3 pin an output pin, while clearing the bit to 0 makes the pin an input pin.

P3DDR is initialized to H'00 (bits 5 to 0) by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode. As the SCI is initialized, the pin states are determined by the P3DDR and P3DR specifications.

**Port 3 Data Register (P3DR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | P35DR | P34DR | P33DR | P32DR | P31DR | P30DR |
| Initial value | : | Undefined | Undefined | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

P3DR is an 8-bit readable/writable register that stores output data for the port 3 pins ($P3_5$ to $P3_0$).

Bits 7 and 6 are reserved; they return an undetermined value if read, and cannot be modified.

P3DR is initialized to H'00 (bits 5 to 0) by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port 3 Register (PORT3)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | P35 | P34 | P33 | P32 | P31 | P30 |
| Initial value | : | Undefined | Undefined | —* | —* | —* | —* | —* | —* |
| R/W | : | — | — | R | R | R | R | R | R |

Note: * Determined by state of pins $P3_5$ to $P3_0$.

PORT3 is an 8-bit read-only register that shows the pin states. Writing of output data for the port 3 pins ($P3_5$ to $P3_0$) must always be performed on P3DR.

Bits 7 and 6 are reserved; they return an undetermined value if read, and cannot be modified.

If a port 3 read is performed while P3DDR bits are set to 1, the P3DR values are read. If a port 3 read is performed while P3DDR bits are cleared to 0, the pin states are read.

After a power-on reset and in hardware standby mode, PORT3 contents are determined by the pin states, as P3DDR and P3DR are initialized. PORT3 retains its prior state after a manual reset, and in software standby mode.

**Port 3 Open Drain Control Register (P3ODR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | P35ODR | P34ODR | P33ODR | P32ODR | P31ODR | P30ODR |
| Initial value | : | Undefined | Undefined | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

P3ODR is an 8-bit readable/writable register that controls the PMOS on/off status for each port 3 pin ($P3_5$ to $P3_0$).

Bits 7 and 6 are reserved; they return an undetermined value if read, and cannot be modified.

Setting a P3ODR bit to 1 makes the corresponding port 3 pin an NMOS open-drain output pin, while clearing the bit to 0 makes the pin a CMOS output pin.

P3ODR is initialized to H'00 (bits 5 to 0) by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

### 8.4.3 Pin Functions

Port 3 pins also function as SCI I/O pins (TxD0, RxD0, SCK0, TxD1, RxD1, and SCK1) and interrupt input pins ($\overline{IRQ4}$, $\overline{IRQ5}$). Port 3 pin functions are shown in table 8.7.

**Table 8.7 Port 3 Pin Functions**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P3_5$/SCK1/$\overline{IRQ5}$ | The pin function is switched as shown below according to the combination of bit C/$\overline{A}$ in the SCI1 SMR, bits CKE0 and CKE1 in SCR, and bit P35DDR. |

| CKE1 | 0 | | | | 1 |
|---|---|---|---|---|---|
| C/$\overline{A}$ | 0 | | | 1 | — |
| CKE0 | 0 | | 1 | — | — |
| P35DDR | 0 | 1 | — | — | — |
| Pin function | $P3_5$ input pin | $P3_5$ output pin*1 | SCK1 output pin*1 | SCK1 output pin*1 | SCK1 input pin |
| | $\overline{IRQ5}$ interrupt input pin*2 | | | | |

Notes: 1. When P35ODR = 1, the pin becomes on NMOS open-drain output.
2. When this pin is used as an external interrupt input, it should not be used as an input/output pin with other functions.

| Pin | Selection Method and Pin Functions |
|---|---|
| $P3_4$/SCK0/$\overline{IRQ4}$ | The pin function is switched as shown below according to the combination of bit C/$\overline{A}$ in the SCI0 SMR, bits CKE0 and CKE1 in SCR, and bit P34DDR. |

| CKE1 | 0 | | | | 1 |
|---|---|---|---|---|---|
| C/$\overline{A}$ | 0 | | | 1 | — |
| CKE0 | 0 | | 1 | — | — |
| P34DDR | 0 | 1 | — | — | — |
| Pin function | $P3_4$ input pin | $P3_4$ output pin*1 | SCK0 output pin*1 | SCK0 output pin*1 | SCK0 input pin |
| | $\overline{IRQ4}$ interrupt input pin*2 | | | | |

Notes: 1. When P34ODR = 1, the pin becomes an NMOS open-drain output.
2. When this pin is used as an external interrupt input, it should not be used as an input/output pin with other functions.

**Table 8.7 Port 3 Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $P3_3$/RxD1 | The pin function is switched as shown below according to the combination of bit RE in the SCI1 SCR, and bit P33DDR. |

| RE | 0 | | 1 |
|---|---|---|---|
| P33DDR | 0 | 1 | — |
| Pin function | $P3_3$ input pin | $P3_3$ output pin* | RxD1 input pin |

Note: * When P33ODR = 1, the pin becomes an NMOS open-drain output.

| | |
|---|---|
| $P3_2$/RxD0 | The pin function is switched as shown below according to the combination of bit RE in the SCI0 SCR, and bit P32DDR. |

| RE | 0 | | 1 |
|---|---|---|---|
| P32DDR | 0 | 1 | — |
| Pin function | $P3_2$ input pin | $P3_2$ output pin* | RxD0 input pin |

Note: * When P32ODR = 1, the pin becomes an NMOS open-drain output.

| | |
|---|---|
| $P3_1$/TxD1 | The pin function is switched as shown below according to the combination of bit TE in the SCI1 SCR, and bit P31DDR. |

| TE | 0 | | 1 |
|---|---|---|---|
| P31DDR | 0 | 1 | — |
| Pin function | $P3_1$ input pin | $P3_1$ output pin* | TxD1 output pin |

Note: * When P31ODR = 1, the pin becomes an NMOS open-drain output.

| | |
|---|---|
| $P3_0$/TxD0 | The pin function is switched as shown below according to the combination of bit TE in the SCI0 SCR, and bit P30DDR. |

| TE | 0 | | 1 |
|---|---|---|---|
| P30DDR | 0 | 1 | — |
| Pin function | $P3_0$ input pin | $P3_0$ output pin* | TxD0 output pin |

Note: * When P30ODR = 1, the pin becomes an NMOS open-drain output.

## 8.5 Port 4

### 8.5.1 Overview

Port 4 is an 8-bit input-only port. Port 4 pins also function as A/D converter analog input pins (AN0 to AN7) and D/A converter analog output pins (DA0 and DA1). Port 4 pin functions are the same in all operating modes. Figure 8.4 shows the port 4 pin configuration.

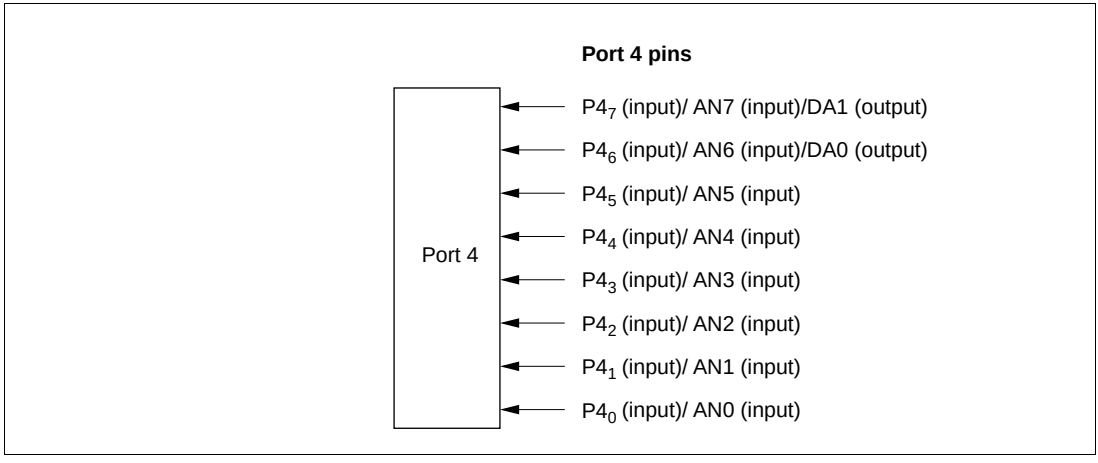



**Figure 8.4 Port 4 Pin Functions**

### 8.5.2 Register Configuration

Table 8.8 shows the port 4 register configuration. Port 4 is an input-only port, and does not have a data direction register or data register.

**Table 8.8 Port 4 Registers**

| Name | Abbreviation | R/W | Initial Value | Address* |
|---|---|---|---|---|
| Port 4 register | PORT4 | R | Undefined | H'FF53 |

Note: * Lower 16 bits of the address.

**Port 4 Register (PORT4):**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | P47 | P46 | P45 | P44 | P43 | P42 | P41 | P40 |
| Initial value | : | —* | —* | —* | —* | —* | —* | —* | —* |
| R/W | : | R | R | R | R | R | R | R | R |

Note: * Determined by state of pins $P4_7$ to $P4_0$.

PORT4 is an 8-bit read-only port. A read always returns the pin states. Writes are invalid.

### 8.5.3 Pin Functions

Port 4 pins also function as A/D converter analog input pins (AN0 to AN7) and D/A converter analog output pins (DA0 and DA1).

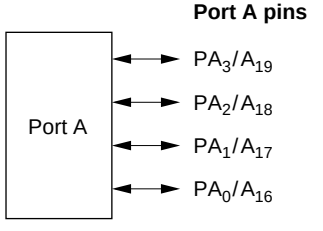
## 8.6 Port A

### 8.6.1 Overview

Port A is an 4-bit I/O port. Port A pins also function as address bus outputs. The pin functions change according to the operating mode.

Port A has a built-in MOS input pull-up function that can be controlled by software.

Figure 8.5 shows the port A pin configuration.




| Port A pins | Pin functions in modes 1, 2, 3, and 7* |
|---|---|
| $PA_3/A_{19}$ | $PA_3$ (I/O) |
| $PA_2/A_{18}$ | $PA_2$ (I/O) |
| $PA_1/A_{17}$ | $PA_1$ (I/O) |
| $PA_0/A_{16}$ | $PA_0$ (I/O) |

| Pin functions in modes 4 and 5 | Pin functions in mode 6* |
|---|---|
| $A_{19}$ (output) | $PA_3$ (input)/$A_{19}$ (output) |
| $A_{18}$ (output) | $PA_2$ (input)/$A_{18}$ (output) |
| $A_{17}$ (output) | $PA_1$ (input)/$A_{17}$ (output) |
| $A_{16}$ (output) | $PA_0$ (input)/$A_{16}$ (output) |

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

**Figure 8.5 Port A Pin Functions**

### 8.6.2 Register Configuration

Table 8.9 shows the port A register configuration.

**Table 8.9 Port A Registers**

| Name | Abbreviation | R/W | Initial Value*[1] | Address*[2] |
|---|---|---|---|---|
| Port A data direction register | PADDR | W | H'0 | H'FEB9 |
| Port A data register | PADR | R/W | H'0 | H'FF69 |
| Port A register | PORTA | R | Undefined | H'FF59 |
| Port A MOS pull-up control register | PAPCR | R/W | H'0 | H'FF70 |
| Port A open-drain control register | PAODR | R/W | H'0 | H'FF77 |

Notes: 1. Value of bits 3 to 0.
2. Lower 16 bits of the address.

**Port A Data Direction Register (PADDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | PA3DDR | PA2DDR | PA1DDR | PA0DDR |
| Initial value | : | Undefined | Undefined | Undefined | Undefined | 0 | 0 | 0 | 0 |
| R/W | : | — | — | — | — | W | W | W | W |

PADDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port A. PADDR cannot be read; if it is, an undefined value will be read. Bits 7 to 4 are reserved.

PADDR is initialized to H'0 (bits 3 to 0) by a power-on reset and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode. The OPE bit in SBYCR is used to select whether the address output pins retain their output state or become high-impedance when a transition is made to software standby mode.

- Modes 1, 2, 3, and 7*

  Setting a PADDR bit to 1 makes the corresponding port A pin an output port, while clearing the bit to 0 makes the pin an input port.

- Modes 4 and 5

  The corresponding port A pins are address outputs irrespective of the value of bits PA3DDR to PA0DDR.

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

- Mode 6*

  Setting a PADDR bit to 1 makes the corresponding port A pin an address output while clearing the bit to 0 makes the pin an input port.

**Port A Data Register (PADR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | PA3DR | PA2DR | PA1DR | PA0DR |
| Initial value | : | Undefined | Undefined | Undefined | Undefined | 0 | 0 | 0 | 0 |
| R/W | : | — | — | — | — | R/W | R/W | R/W | R/W |

PADR is an 8-bit readable/writable register that stores output data for the port A pins ($PA_3$ to $PA_0$).

Bits 7 to 4 are reserved; they return an undetermined value if read, and cannot be modified.

PADR is initialized to H'0 (bits 3 to 0) by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port A Register (PORTA)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | PA3 | PA2 | PA1 | PA0 |
| Initial value | : | Undefined | Undefined | Undefined | Undefined | —* | —* | —* | —* |
| R/W | : | — | — | — | — | R | R | R | R |

Note: * Determined by state of pins $PA_3$ to $PA_0$.

PORTA is an 8-bit read-only register that shows the pin states. It cannot be written to. Writing of output data for the port A pins ($PA_3$ to $PA_0$) must always be performed on PADR.

Bits 7 to 4 are reserved; they return an undetermined value if read, and cannot be modified.

If a port A read is performed while PADDR bits are set to 1, the PADR values are read. If a port A read is performed while PADDR bits are cleared to 0, the pin states are read.

After a power-on reset and in hardware standby mode, PORTA contents are determined by the pin states, as PADDR and PADR are initialized. PORTA retains its prior state after a manual reset, and in software standby mode.

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

**Port A MOS Pull-Up Control Register (PAPCR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | PA3PCR | PA2PCR | PA1PCR | PA0PCR |
| Initial value | : | Undefined | Undefined | Undefined | Undefined | 0 | 0 | 0 | 0 |
| R/W | : | — | — | — | — | R/W | R/W | R/W | R/W |

PAPCR is an 8-bit readable/writable register that controls the MOS input pull-up function incorporated into port A on an individual bit basis.

Bits 7 to 4 are reserved; they return an undetermined value if read, and cannot be modified.

Bits 3 to 0 are valid in modes 1, 2, 3, 6, and 7, and all the bits are invalid in modes 4 and 5. When a PADDR bit is cleared to 0 (input port setting), setting the corresponding PAPCR bit to 1 turns on the MOS input pull-up for the corresponding pin.

PAPCR is initialized to H'0 (bits 3 to 0) by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port A Open Drain Control Register (PAODR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | PA3ODR | PA2ODR | PA1ODR | PA0ODR |
| Initial value | : | Undefined | Undefined | Undefined | Undefined | 0 | 0 | 0 | 0 |
| R/W | : | — | — | — | — | R/W | R/W | R/W | R/W |

PAODR is an 8-bit readable/writable register that controls whether PMOS is on or off for each port A pin ($PA_3$ to $PA_0$).

Bits 7 to 4 are reserved; they return an undetermined value if read, and cannot be modified.

All bits are valid in modes 1, 2, 3, and 7.*

Setting a PAODR bit to 1 makes the corresponding port A pin an NMOS open-drain output, while clearing the bit to 0 makes the pin a CMOS output.

PAODR is initialized to H'0 (bits 3 to 0) by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

### 8.6.3 Pin Functions

**Modes 1, 2, 3 and 7*:** In mode 1, 2, 3, and 7*, port A pins function as I/O ports. Input or output can be specified for each pin on an individual bit basis. Setting a PADDR bit to 1 makes the corresponding port A pin an output port, while clearing the bit to 0 makes the pin an input port.

Port A pin functions in modes 1, 2, 3, and 7 are shown in figure 8.6.

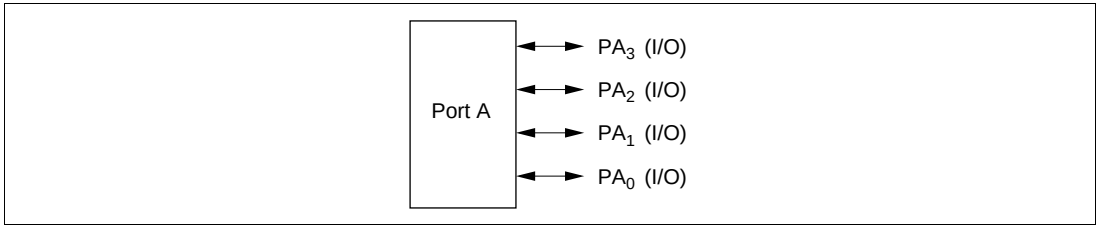



**Figure 8.6 Port A Pin Functions (Modes 1, 2, 3, and 7)***

**Modes 4 and 5:** In modes 4 and 5, the lower 4 bits of port A are designated as address outputs automatically.

Port A pin functions in modes 4 and 5 are shown in figure 8.7.

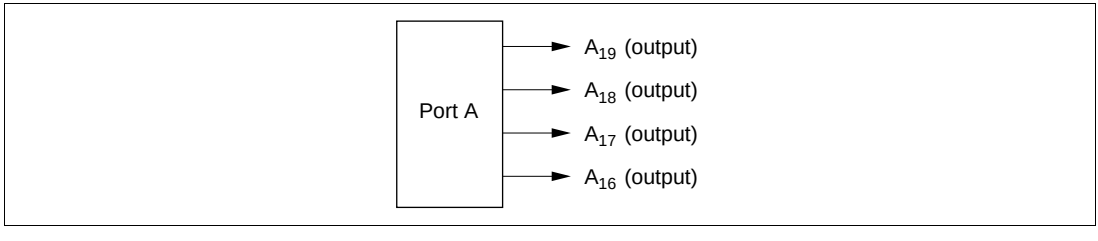



**Figure 8.7 Port A Pin Functions (Modes 4 and 5)**

**Mode 6*:** In mode 6*, port A pins function as address outputs or input ports. Input or output can be specified on an individual bit basis. Setting a PADDR bit to 1 makes the corresponding port A pin an address output, while clearing the bit to 0 makes the pin an input port.

Port A pin functions in mode 6 are shown in figure 8.8.

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

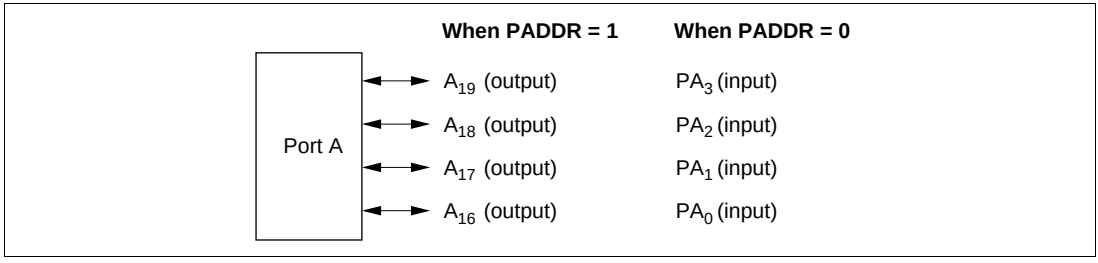



**Figure 8.8 Port A Pin Functions (Mode 6)***

### 8.6.4 MOS Input Pull-Up Function

Port A has a built-in MOS input pull-up function that can be controlled by software. This MOS input pull-up function can be used in modes 1, 2, 3, 6, and 7*, and cannot be used in modes 4 and 5. MOS input pull-up can be specified as on or off on an individual bit basis.

When a PADDR bit is cleared to 0, setting the corresponding PAPCR bit to 1 turns on the MOS input pull-up for that pin.

The MOS input pull-up function is in the off state after a power-on reset, and in hardware standby mode. The prior state is retained after a manual reset, and in software standby mode.

Table 8.10 summarizes the MOS input pull-up states.

**Table 8.10 MOS Input Pull-Up States (Port A)**

| Modes | | Power-On Reset | Hardware Standby Mode | Manual Reset | Software Standby Mode | In Other Operations |
|---|---|---|---|---|---|---|
| 1 to 3, 6, 7* | $PA_3$ to $PA_0$ | OFF | | ON/OFF | | |
| 4, 5 | $PA_3$ to $PA_0$ | | | OFF | | |

**Legend:**

OFF : MOS input pull-up is always off.

ON/OFF : On when PADDR = 0 and PAPCR = 1; otherwise off.

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

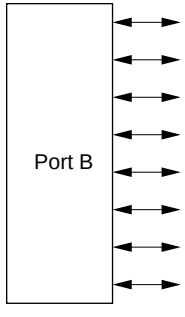
## 8.7 Port B

### 8.7.1 Overview

Port B is an 8-bit I/O port. Port B has an address bus output function, and the pin functions change according to the operating mode.

Port B has a built-in MOS input pull-up function that can be controlled by software.

Figure 8.9 shows the port B pin configuration.




| Port B pins | Pin functions in modes 1, 4, and 5* |
|---|---|
| $PB_7/A_{15}$ | $A_{15}$ (output) |
| $PB_6/A_{14}$ | $A_{14}$ (output) |
| $PB_5/A_{13}$ | $A_{13}$ (output) |
| $PB_4/A_{12}$ | $A_{12}$ (output) |
| $PB_3/A_{11}$ | $A_{11}$ (output) |
| $PB_2/A_{10}$ | $A_{10}$ (output) |
| $PB_1/A_9$ | $A_9$ (output) |
| $PB_0/A_8$ | $A_8$ (output) |

| Pin functions in modes 2 and 6* | Pin functions in modes 3 and 7* |
|---|---|
| $PB_7$ (input)/$A_{15}$ (output) | $PB_7$ (I/O) |
| $PB_6$ (input)/$A_{14}$ (output) | $PB_6$ (I/O) |
| $PB_5$ (input)/$A_{13}$ (output) | $PB_5$ (I/O) |
| $PB_4$ (input)/$A_{12}$ (output) | $PB_4$ (I/O) |
| $PB_3$ (input)/$A_{11}$ (output) | $PB_3$ (I/O) |
| $PB_2$ (input)/$A_{10}$ (output) | $PB_2$ (I/O) |
| $PB_1$ (input)/$A_9$ (output) | $PB_1$ (I/O) |
| $PB_0$ (input)/$A_8$ (output) | $PB_0$ (I/O) |

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

**Figure 8.9 Port B Pin Functions**

### 8.7.2 Register Configuration

Table 8.11 shows the port B register configuration.

**Table 8.11 Port B Registers**

| Name | Abbreviation | R/W | Initial Value | Address * |
|---|---|---|---|---|
| Port B data direction register | PBDDR | W | H'00 | H'FEBA |
| Port B data register | PBDR | R/W | H'00 | H'FF6A |
| Port B register | PORTB | R | Undefined | H'FF5A |
| Port B MOS pull-up control register | PBPCR | R/W | H'00 | H'FF71 |

Note: * Lower 16 bits of the address.

**Port B Data Direction Register (PBDDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PB7DDR | PB6DDR | PB5DDR | PB4DDR | PB3DDR | PB2DDR | PB1DDR | PB0DDR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | W | W | W | W | W | W | W | W |

PBDDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port B. PBDDR cannot be read; if it is, an undefined value will be read.

PBDDR is initialized to H'00 by a power-on reset and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode. The OPE bit in SBYCR is used to select whether the address output pins retain their output state or become high-impedance when a transition is made to software standby mode.

- Modes 1, 4, and 5*

  The corresponding port B pins are address outputs irrespective of the value of the PBDDR bits.

- Modes 2 and 6*

  Setting a PBDDR bit to 1 makes the corresponding port B pin an address output, while clearing the bit to 0 makes the pin an input port.

- Modes 3 and 7*

  Setting a PBDDR bit to 1 makes the corresponding port B pin an output port, while clearing the bit to 0 makes the pin an input port.

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

**Port B Data Register (PBDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PB7DR | PB6DR | PB5DR | PB4DR | PB3DR | PB2DR | PB1DR | PB0DR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PBDR is an 8-bit readable/writable register that stores output data for the port B pins ($PB_7$ to $PB_0$). PBDR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port B Register (PORTB)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PB7 | PB6 | PB5 | PB4 | PB3 | PB2 | PB1 | PB0 |
| Initial value | : | —* | —* | —* | —* | —* | —* | —* | —* |
| R/W | : | R | R | R | R | R | R | R | R |

Note: * Determined by state of pins $PB_7$ to $PB_0$.

PORTB is an 8-bit read-only register that shows the pin states. It cannot be written to. Writing of output data for the port B pins ($PB_7$ to $PB_0$) must always be performed on PBDR.

If a port B read is performed while PBDDR bits are set to 1, the PBDR values are read. If a port B read is performed while PBDDR bits are cleared to 0, the pin states are read.

After a power-on reset and in hardware standby mode, PORTB contents are determined by the pin states, as PBDDR and PBDR are initialized. PORTB retains its prior state after a manual reset, and in software standby mode.

**Port B MOS Pull-Up Control Register (PBPCR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PB7PCR | PB6PCR | PB5PCR | PB4PCR | PB3PCR | PB2PCR | PB1PCR | PB0PCR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PBPCR is an 8-bit readable/writable register that controls the MOS input pull-up function incorporated into port B on an individual bit basis.

When a PBDDR bit is cleared to 0 (input port setting) in mode 2, 3, 6, or 7*, setting the corresponding PBPCR bit to 1 turns on the MOS input pull-up for the corresponding pin.

PBPCR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

### 8.7.3 Pin Functions

**Modes 1, 4, and 5*:** In modes 1, 4, and 5*, port B pins are automatically designated as address outputs.

Port B pin functions in modes 1, 4, and 5 are shown in figure 8.10.

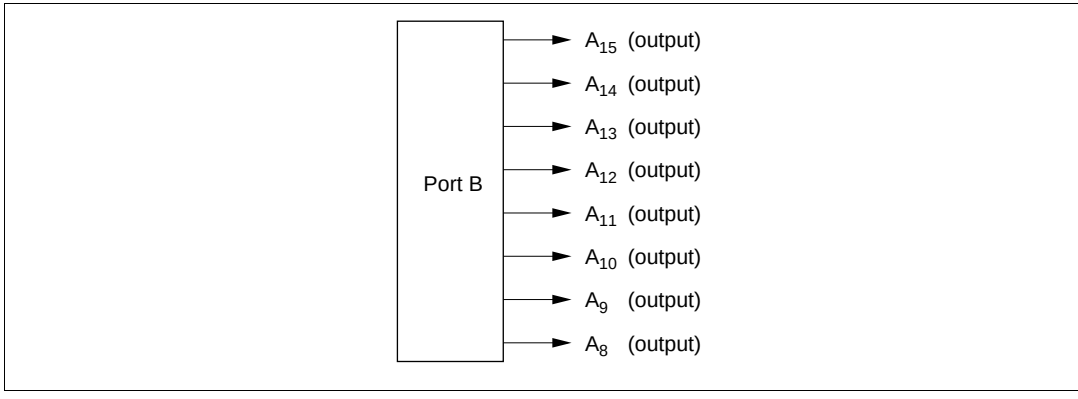



**Figure 8.10 Port B Pin Functions (Modes 1, 4, and 5)***

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

**Modes 2 and 6*:** In modes 2 and 6*, port B pins function as address outputs or input ports. Input or output can be specified on an individual bit basis. Setting a PBDDR bit to 1 makes the corresponding port B pin an address output, while clearing the bit to 0 makes the pin an input port.

Port B pin functions in modes 2 and 6 are shown in figure 8.11.

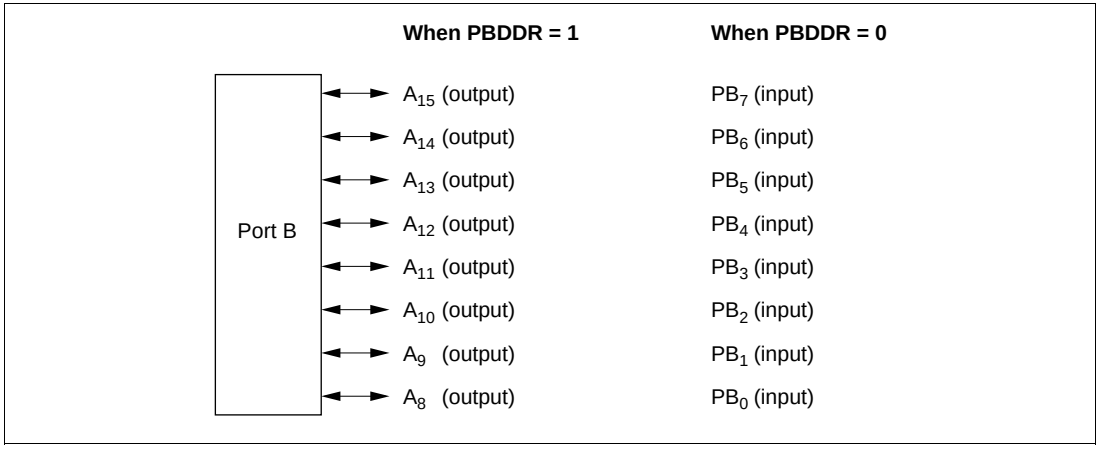



**Figure 8.11 Port B Pin Functions (Modes 2 and 6) ***

**Modes 3 and 7*:** In modes 3 and 7*, port B pins function as I/O ports. Input or output can be specified for each pin on an individual bit basis. Setting a PBDDR bit to 1 makes the corresponding port B pin an output port, while clearing the bit to 0 makes the pin an input port.

Port B pin functions in modes 3 and 7 are shown in figure 8.12.

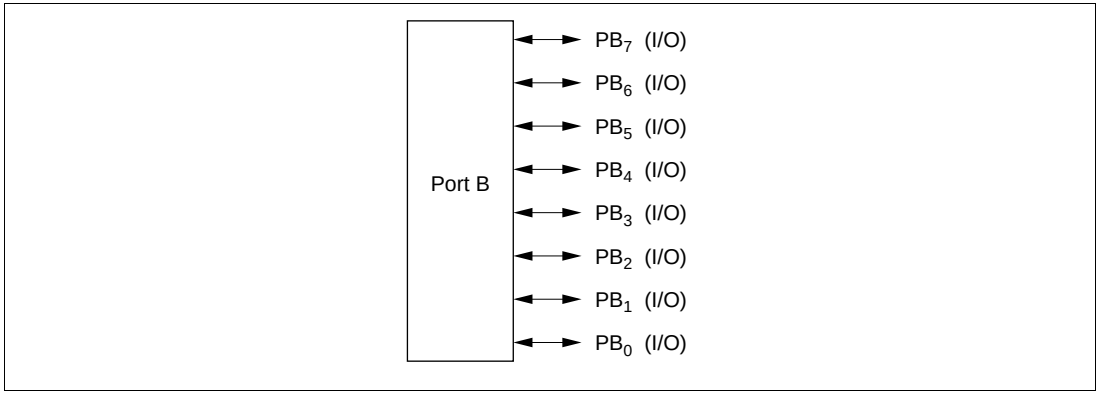



**Figure 8.12 Port B Pin Functions (Modes 3 and 7)***

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

### 8.7.4 MOS Input Pull-Up Function

Port B has a built-in MOS input pull-up function that can be controlled by software. This MOS input pull-up function can be used in modes 2, 3, 6, and 7, and can be specified as on or off on an individual bit basis.

When a PBDDR bit is cleared to 0 in mode 2, 3, 6, or 7, setting the corresponding PBPCR bit to 1 turns on the MOS input pull-up for that pin.

The MOS input pull-up function is in the off state after a power-on reset, and in hardware standby mode. The prior state is retained after a manual reset, and in software standby mode.

Table 8.12 summarizes the MOS input pull-up states.

**Table 8.12 MOS Input Pull-Up States (Port B)**

| Modes | Power-On Reset | Hardware Standby Mode | Manual Reset | Software Standby Mode | In Other Operations |
|---|---|---|---|---|---|
| 1, 4, 5* | OFF | | OFF | | |
| 2, 3, 6, 7* | | | ON/OFF | | |

**Legend:**

OFF : MOS input pull-up is always off.

ON/OFF : On when PBDDR = 0 and PBPCR = 1; otherwise off.

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

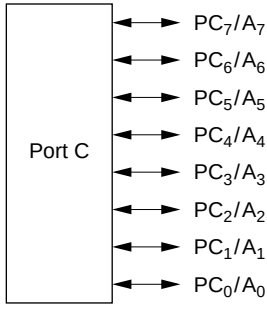
## 8.8 Port C

### 8.8.1 Overview

Port C is an 8-bit I/O port. Port C has an address bus output function, and the pin functions change according to the operating mode.

Port C has a built-in MOS input pull-up function that can be controlled by software.

Figure 8.13 shows the port C pin configuration.




**Pin functions in modes 1, 4, and 5***

| | |
|---|---|
| $PC_7/A_7$ | $A_7$ (output) |
| $PC_6/A_6$ | $A_6$ (output) |
| $PC_5/A_5$ | $A_5$ (output) |
| $PC_4/A_4$ | $A_4$ (output) |
| $PC_3/A_3$ | $A_3$ (output) |
| $PC_2/A_2$ | $A_2$ (output) |
| $PC_1/A_1$ | $A_1$ (output) |
| $PC_0/A_0$ | $A_0$ (output) |

| **Pin functions in modes 2 and 6*** | **Pin functions in modes 3 and 7*** |
|---|---|
| $PC_7$ (input)/ $A_7$ (output) | $PC_7$ (I/O) |
| $PC_6$ (input)/ $A_6$ (output) | $PC_6$ (I/O) |
| $PC_5$ (input)/ $A_5$ (output) | $PC_5$ (I/O) |
| $PC_4$ (input)/ $A_4$ (output) | $PC_4$ (I/O) |
| $PC_3$ (input)/ $A_3$ (output) | $PC_3$ (I/O) |
| $PC_2$ (input)/ $A_2$ (output) | $PC_2$ (I/O) |
| $PC_1$ (input)/ $A_1$ (output) | $PC_1$ (I/O) |
| $PC_0$ (input)/ $A_0$ (output) | $PC_0$ (I/O) |

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

**Figure 8.13 Port C Pin Functions**

### 8.8.2 Register Configuration

Table 8.13 shows the port C register configuration.

**Table 8.13 Port C Registers**

| Name | Abbreviation | R/W | Initial Value | Address * |
|---|---|---|---|---|
| Port C data direction register | PCDDR | W | H'00 | H'FEBB |
| Port C data register | PCDR | R/W | H'00 | H'FF6B |
| Port C register | PORTC | R | Undefined | H'FF5B |
| Port C MOS pull-up control register | PCPCR | R/W | H'00 | H'FF72 |

Note: * Lower 16 bits of the address.

**Port C Data Direction Register (PCDDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PC7DDR | PC6DDR | PC5DDR | PC4DDR | PC3DDR | PC2DDR | PC1DDR | PC0DDR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | W | W | W | W | W | W | W | W |

PCDDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port C. PCDDR cannot be read; if it is, an undefined value will be read.

PCDDR is initialized to H'00 by a power-on reset and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode. The OPE bit in SBYCR is used to select whether the address output pins retain their output state or become high-impedance when a transition is made to software standby mode.

- Modes 1, 4, and 5*

  The corresponding port C pins are address outputs irrespective of the value of the PCDDR bits.

- Modes 2 and 6*

  Setting a PCDDR bit to 1 makes the corresponding port C pin an address output, while clearing the bit to 0 makes the pin an input port.

- Modes 3 and 7*

  Setting a PCDDR bit to 1 makes the corresponding port C pin an output port, while clearing the bit to 0 makes the pin an input port.

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

**Port C Data Register (PCDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PC7DR | PC6DR | PC5DR | PC4DR | PC3DR | PC2DR | PC1DR | PC0DR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PCDR is an 8-bit readable/writable register that stores output data for the port C pins ($PC_7$ to $PC_0$).

PCDR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port C Register (PORTC)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PC7 | PC6 | PC5 | PC4 | PC3 | PC2 | PC1 | PC0 |
| Initial value | : | —* | —* | —* | —* | —* | —* | —* | —* |
| R/W | : | R | R | R | R | R | R | R | R |

Note: * Determined by state of pins $PC_7$ to $PC_0$.

PORTC is an 8-bit read-only register that shows the pin states. It cannot be written to. Writing of output data for the port C pins ($PC_7$ to $PC_0$) must always be performed on PCDR.

If a port C read is performed while PCDDR bits are set to 1, the PCDR values are read. If a port C read is performed while PCDDR bits are cleared to 0, the pin states are read.

After a power-on reset and in hardware standby mode, PORTC contents are determined by the pin states, as PCDDR and PCDR are initialized. PORTC retains its prior state after a manual reset, and in software standby mode.

**Port C MOS Pull-Up Control Register (PCPCR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PC7PCR | PC6PCR | PC5PCR | PC4PCR | PC3PCR | PC2PCR | PC1PCR | PC0PCR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PCPCR is an 8-bit readable/writable register that controls the MOS input pull-up function incorporated into port C on an individual bit basis.

When a PCDDR bit is cleared to 0 (input port setting) in mode 2, 3, 6, or 7*, setting the corresponding PCPCR bit to 1 turns on the MOS input pull-up for the corresponding pin.

PCPCR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

### 8.8.3 Pin Functions

**Modes 1, 4, and 5*:** In modes 1, 4, and 5*, port C pins are automatically designated as address outputs.

Port C pin functions in modes 1, 4, and 5 are shown in figure 8.14.

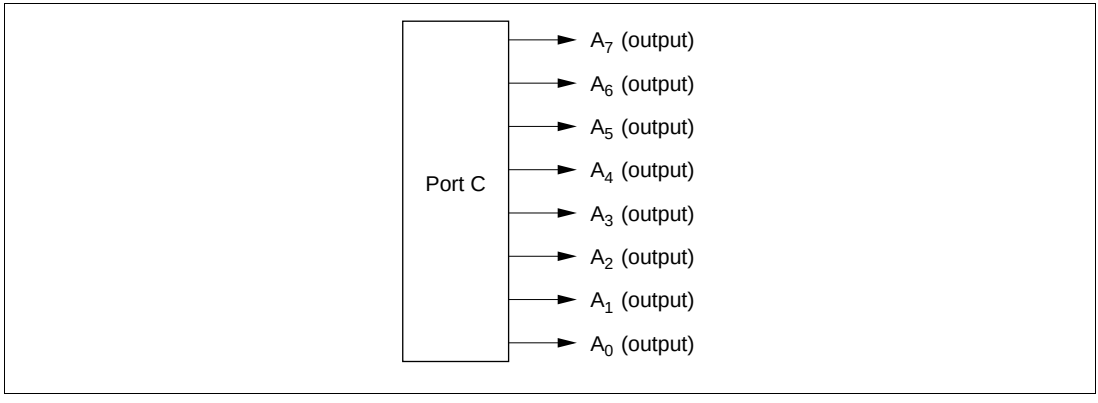



**Figure 8.14 Port C Pin Functions (Modes 1, 4, and 5)***

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

**Modes 2 and 6*:** In modes 2 and 6*, port C pins function as address outputs or input ports. Input or output can be specified on an individual bit basis. Setting a PCDDR bit to 1 makes the corresponding port C pin an address output, while clearing the bit to 0 makes the pin an input port.

Port C pin functions in modes 2 and 6 are shown in figure 8.15.

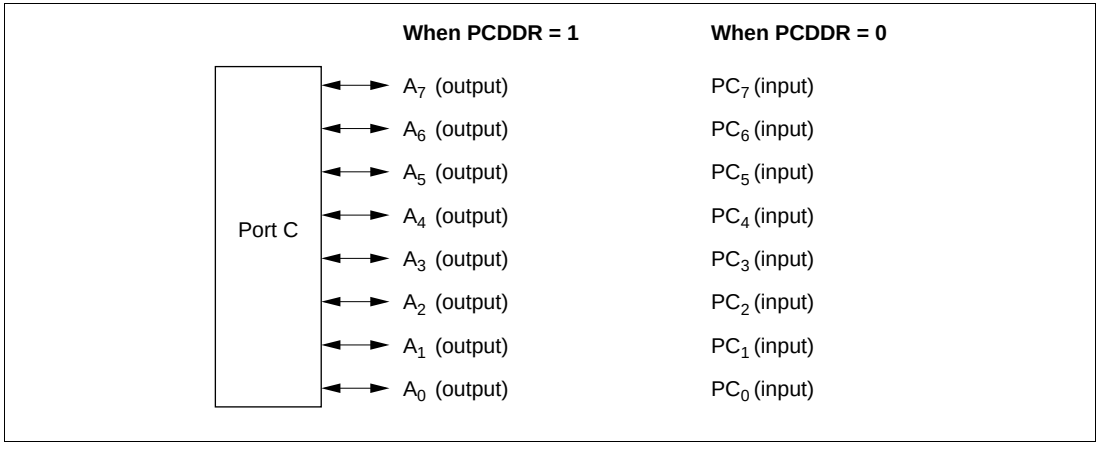



**Figure 8.15 Port C Pin Functions (Modes 2 and 6)***

**Modes 3 and 7*:** In modes 3 and 7*, port C pins function as I/O ports. Input or output can be specified for each pin on an individual bit basis. Setting a PCDDR bit to 1 makes the corresponding port C pin an output port, while clearing the bit to 0 makes the pin an input port.

Port C pin functions in modes 3 and 7 are shown in figure 8.16.

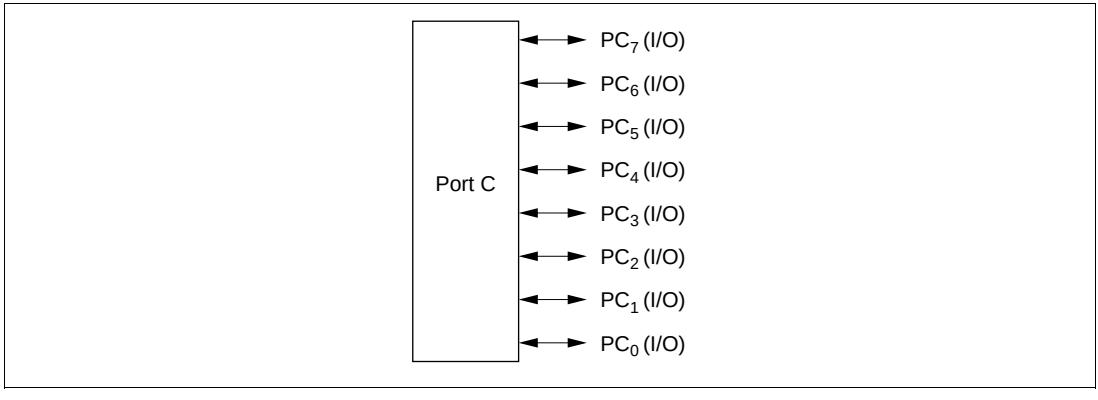



**Figure 8.16 Port C Pin Functions (Modes 3 and 7)***

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

### 8.8.4 MOS Input Pull-Up Function

Port C has a built-in MOS input pull-up function that can be controlled by software. This MOS input pull-up function can be used in modes 2, 3, 6, and 7*, and can be specified as on or off on an individual bit basis.

When a PCDDR bit is cleared to 0 in mode 2, 3, 6, or 7*, setting the corresponding PCPCR bit to 1 turns on the MOS input pull-up for that pin.

The MOS input pull-up function is in the off state after a power-on reset, and in hardware standby mode. The prior state is retained after a manual reset, and in software standby mode.

Table 8.14 summarizes the MOS input pull-up states.

**Table 8.14 MOS Input Pull-Up States (Port C)**

| Modes | Power-On Reset | Hardware Standby Mode | Manual Reset | Software Standby Mode | In Other Operations |
|---|---|---|---|---|---|
| 1, 4, 5* | OFF | | OFF | | |
| 2, 3, 6, 7* | | | ON/OFF | | |

**Legend:**

OFF : MOS input pull-up is always off.

ON/OFF : On when PCDDR = 0 and PCPCR = 1; otherwise off.

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

## 8.9 Port D

### 8.9.1 Overview

Port D is an 8-bit I/O port. Port D has a data bus I/O function, and the pin functions change according to the operating mode.

Port D has a built-in MOS input pull-up function that can be controlled by software.

Figure 8.17 shows the port D pin configuration.

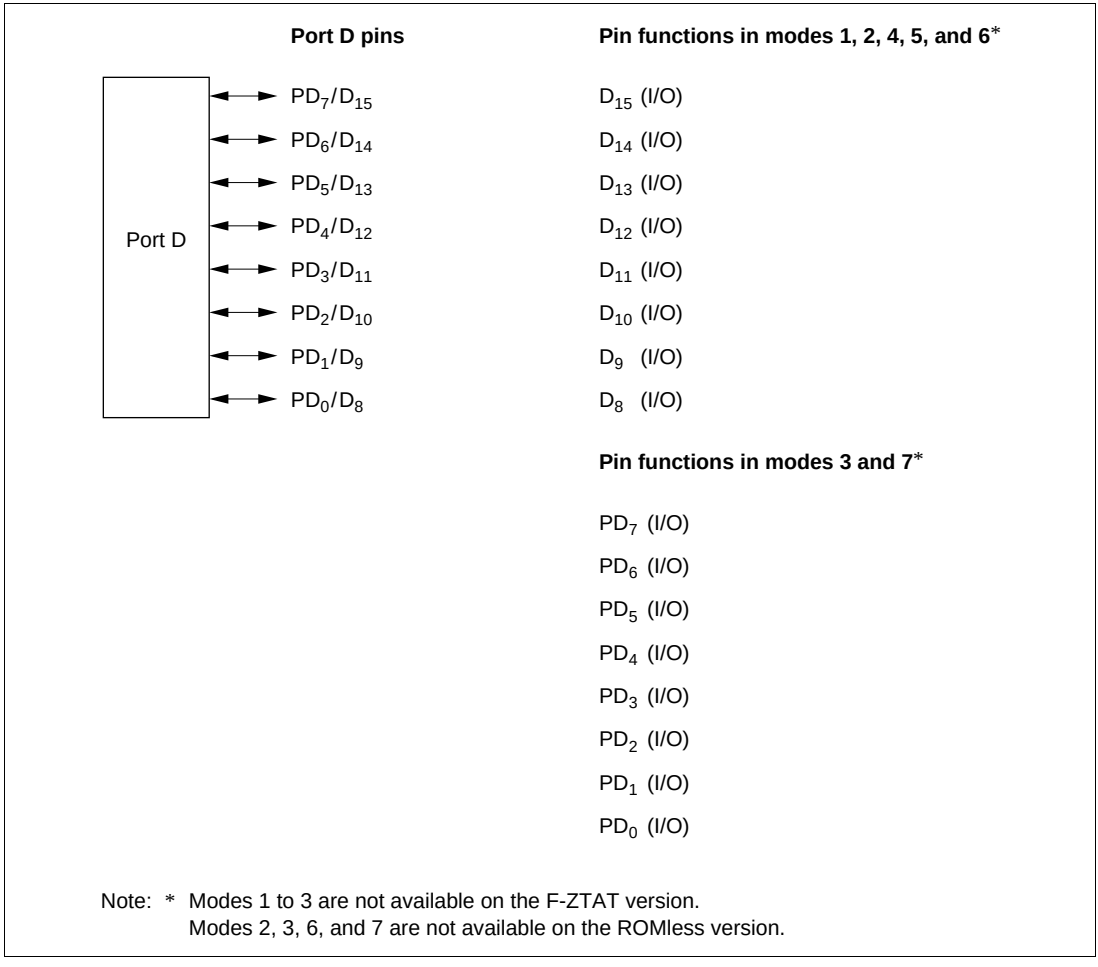



**Figure 8.17 Port D Pin Functions**

### 8.9.2 Register Configuration

Table 8.15 shows the port D register configuration.

**Table 8.15 Port D Registers**

| Name | Abbreviation | R/W | Initial Value | Address * |
|---|---|---|---|---|
| Port D data direction register | PDDDR | W | H'00 | H'FEBC |
| Port D data register | PDDR | R/W | H'00 | H'FF6C |
| Port D register | PORTD | R | Undefined | H'FF5C |
| Port D MOS pull-up control register | PDPCR | R/W | H'00 | H'FF73 |

Note: * Lower 16 bits of the address.

**Port D Data Direction Register (PDDDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PD7DDR | PD6DDR | PD5DDR | PD4DDR | PD3DDR | PD2DDR | PD1DDR | PD0DDR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | W | W | W | W | W | W | W | W |

PDDDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port D. PDDDR cannot be read; if it is, an undefined value will be read..

PDDDR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

- Modes 1, 2, 4, 5, and 6*

  The input/output direction specification by PDDDR is ignored, and port D is automatically designated for data I/O.

- Modes 3 and 7*

  Setting a PDDDR bit to 1 makes the corresponding port D pin an output port, while clearing the bit to 0 makes the pin an input port.

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

**Port D Data Register (PDDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PD7DR | PD6DR | PD5DR | PD4DR | PD3DR | PD2DR | PD1DR | PD0DR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PDDR is an 8-bit readable/writable register that stores output data for the port D pins ($PD_7$ to $PD_0$).

PDDR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port D Register (PORTD)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PD7 | PD6 | PD5 | PD4 | PD3 | PD2 | PD1 | PD0 |
| Initial value | : | —* | —* | —* | —* | —* | —* | —* | —* |
| R/W | : | R | R | R | R | R | R | R | R |

Note: * Determined by state of pins $PD_7$ to $PD_0$.

PORTD is an 8-bit read-only register that shows the pin states. It cannot be written to. Writing of output data for the port D pins ($PD_7$ to $PD_0$) must always be performed on PDDR.

If a port D read is performed while PDDDR bits are set to 1, the PDDR values are read. If a port D read is performed while PDDDR bits are cleared to 0, the pin states are read.

After a power-on reset and in hardware standby mode, PORTD contents are determined by the pin states, as PDDDR and PDDR are initialized. PORTD retains its prior state after a manual reset, and in software standby mode.

**Port D MOS Pull-Up Control Register (PDPCR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PD7PCR | PD6PCR | PD5PCR | PD4PCR | PD3PCR | PD2PCR | PD1PCR | PD0PCR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PDPCR is an 8-bit readable/writable register that controls the MOS input pull-up function incorporated into port D on an individual bit basis.

When a PDDDR bit is cleared to 0 (input port setting) in mode 3 or 7, setting the corresponding PDPCR bit to 1 turns on the MOS input pull-up for the corresponding pin.

PDPCR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

### 8.9.3 Pin Functions

**Modes 1, 2, 4, 5, and 6*:** In modes 1, 2, 4, 5, and 6*, port D pins are automatically designated as data I/O pins.

Port D pin functions in modes 1, 2, 4, 5, and 6 are shown in figure 8.18.

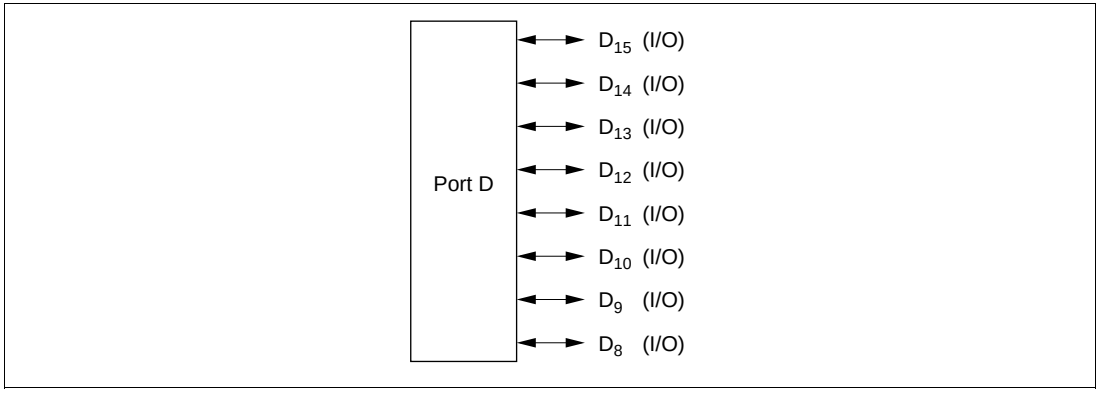



**Figure 8.18 Port D Pin Functions (Modes 1, 2, 4, 5, and 6)***

**Modes 3 and 7*:** In modes 3 and 7*, port D pins function as I/O ports. Input or output can be specified for each pin on an individual bit basis. Setting a PDDDR bit to 1 makes the corresponding port D pin an output port, while clearing the bit to 0 makes the pin an input port.

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

Port D pin functions in modes 3 and 7 are shown in figure 8.19.

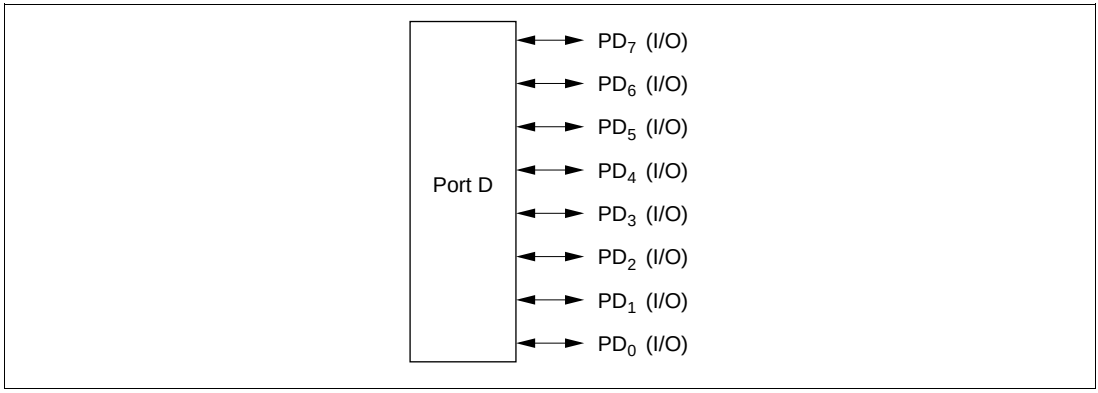



**Figure 8.19 Port D Pin Functions (Modes 3 and 7)***

### 8.9.4 MOS Input Pull-Up Function

Port D has a built-in MOS input pull-up function that can be controlled by software. This MOS input pull-up function can be used in modes 3 and 7*, and can be specified as on or off on an individual bit basis.

When a PDDDR bit is cleared to 0 in mode 3 or 7*, setting the corresponding PDPCR bit to 1 turns on the MOS input pull-up for that pin.

The MOS input pull-up function is in the off state after a power-on reset, and in hardware standby mode. The prior state is retained after a manual reset, and in software standby mode.

Table 8.16 summarizes the MOS input pull-up states.

**Table 8.16 MOS Input Pull-Up States (Port D)**

| Modes | Power-On Reset | Hardware Standby Mode | Manual Reset | Software Standby Mode | In Other Operations |
|---|---|---|---|---|---|
| 1, 2, 4 to 6* | OFF | | OFF | | |
| 3, 7* | | | ON/OFF | | |

**Legend:**

OFF : MOS input pull-up is always off.

ON/OFF : On when PDDDR = 0 and PDPCR = 1; otherwise off.

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

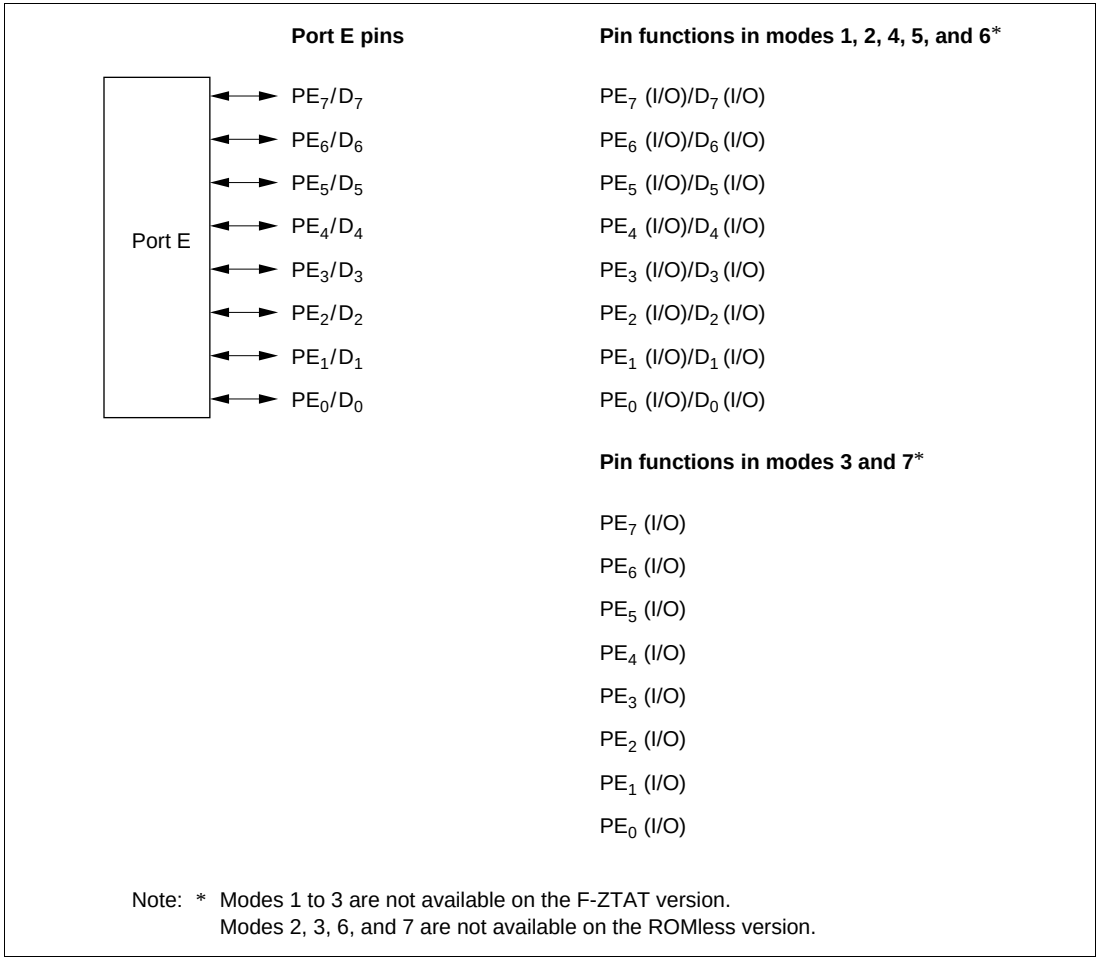
## 8.10 Port E

### 8.10.1 Overview

Port E is an 8-bit I/O port. Port E has a data bus I/O function, and the pin functions change according to the operating mode and whether 8-bit or 16-bit bus mode is selected.

Port E has a built-in MOS input pull-up function that can be controlled by software.

Figure 8.20 shows the port E pin configuration.




Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

**Figure 8.20 Port E Pin Functions**

### 8.10.2 Register Configuration

Table 8.17 shows the port E register configuration.

**Table 8.17 Port E Registers**

| Name | Abbreviation | R/W | Initial Value | Address * |
|---|---|---|---|---|
| Port E data direction register | PEDDR | W | H'00 | H'FEBD |
| Port E data register | PEDR | R/W | H'00 | H'FF6D |
| Port E register | PORTE | R | Undefined | H'FF5D |
| Port E MOS pull-up control register | PEPCR | R/W | H'00 | H'FF74 |

Note: * Lower 16 bits of the address.

**Port E Data Direction Register (PEDDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PE7DDR | PE6DDR | PE5DDR | PE4DDR | PE3DDR | PE2DDR | PE1DDR | PE0DDR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | W | W | W | W | W | W | W | W |

PEDDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port E. PEDDR cannot be read; if it is, an undefined value will be read.

PEDDR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

- Modes 1, 2, 4, 5, and 6*

  When 8-bit bus mode has been selected, port E pins function as I/O ports. Setting a PEDDR bit to 1 makes the corresponding port E pin an output port, while clearing the bit to 0 makes the pin an input port.

  When 16-bit bus mode has been selected, the input/output direction specification by PEDDR is ignored, and port E is designated for data I/O.

  For details of 8-bit and 16-bit bus modes, see section 6, Bus Controller.

- Modes 3 and 7*

  Setting a PEDDR bit to 1 makes the corresponding port E pin an output port, while clearing the bit to 0 makes the pin an input port.

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

**Port E Data Register (PEDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PE7DR | PE6DR | PE5DR | PE4DR | PE3DR | PE2DR | PE1DR | PE0DR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PEDR is an 8-bit readable/writable register that stores output data for the port E pins ($PE_7$ to $PE_0$).

PEDR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port E Register (PORTE)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PE7 | PE6 | PE5 | PE4 | PE3 | PE2 | PE1 | PE0 |
| Initial value | : | —* | —* | —* | —* | —* | —* | —* | —* |
| R/W | : | R | R | R | R | R | R | R | R |

Note: * Determined by state of pins $PE_7$ to $PE_0$.

PORTE is an 8-bit read-only register that shows the pin states. It cannot be written to. Writing of output data for the port E pins ($PE_7$ to $PE_0$) must always be performed on PEDR.

If a port E read is performed while PEDDR bits are set to 1, the PEDR values are read. If a port E read is performed while PEDDR bits are cleared to 0, the pin states are read.

After a power-on reset and in hardware standby mode, PORTE contents are determined by the pin states, as PEDDR and PEDR are initialized. PORTE retains its prior state after a manual reset, and in software standby mode.

**Port E MOS Pull-Up Control Register (PEPCR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PE7PCR | PE6PCR | PE5PCR | PE4PCR | PE3PCR | PE2PCR | PE1PCR | PE0PCR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PEPCR is an 8-bit readable/writable register that controls the MOS input pull-up function incorporated into port E on an individual bit basis.

When a PEDDR bit is cleared to 0 (input port setting) when 8-bit bus mode is selected in mode 1, 2, 4, 5, or 6*, or in mode 3 or 7*, setting the corresponding PEPCR bit to 1 turns on the MOS input pull-up for the corresponding pin.

PEPCR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

### 8.10.3 Pin Functions

**Modes 1, 2, 4, 5, and 6*:** In modes 1, 2, 4, 5, and 6*, when 8-bit access is designated and 8-bit bus mode is selected, port E pins are automatically designated as I/O ports. Setting a PEDDR bit to 1 makes the corresponding port E pin an output port, while clearing the bit to 0 makes the pin an input port.

When 16-bit bus mode is selected, the input/output direction specification by PEDDR is ignored, and port E is designated for data I/O.

Port E pin functions in modes 1, 2, 4, 5, and 6 are shown in figure 8.21.

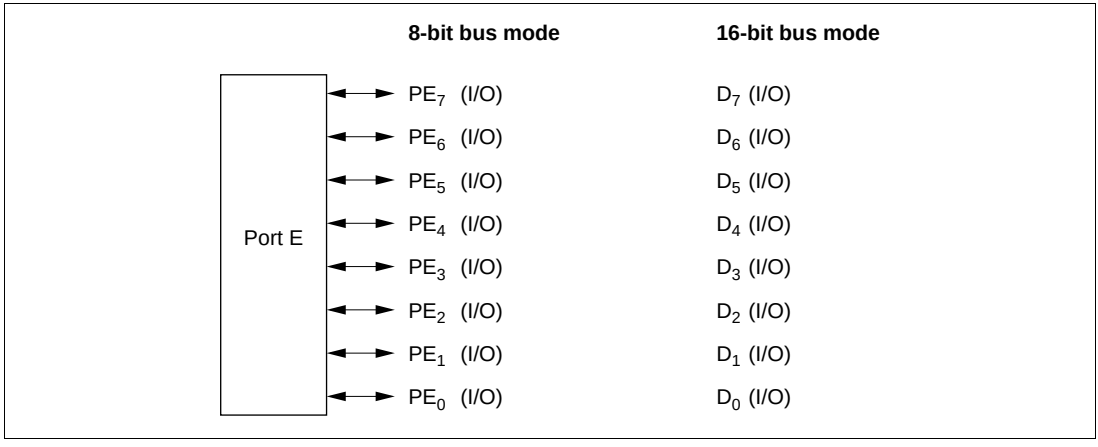



**Figure 8.21 Port E Pin Functions (Modes 1, 2, 4, 5, and 6)***

**Modes 3 and 7*:** In modes 3 and 7*, port E pins function as I/O ports. Input or output can be specified for each pin on a bit-by-bit basis. Setting a PEDDR bit to 1 makes the corresponding port E pin an output port, while clearing the bit to 0 makes the pin an input port.

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

Port E pin functions in modes 3 and 7 are shown in figure 8.22.

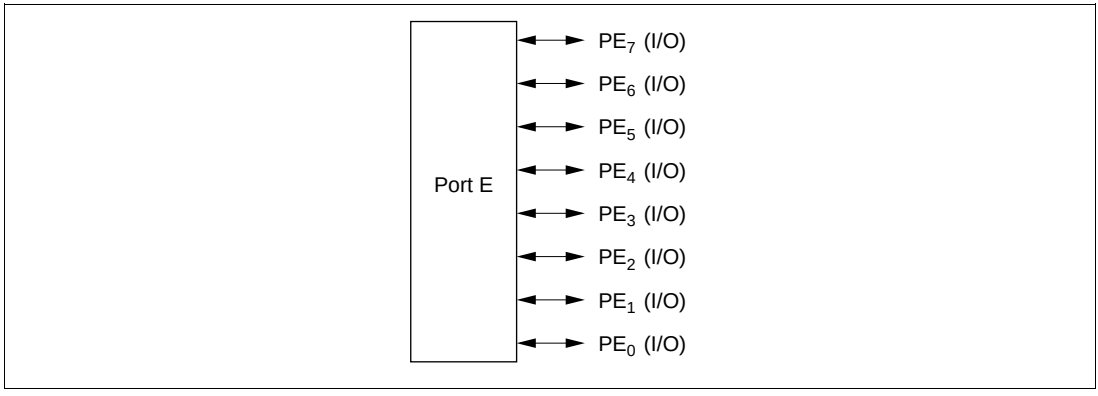



**Figure 8.22 Port E Pin Functions (Modes 3 and 7)***

### 8.10.4 MOS Input Pull-Up Function

Port E has a built-in MOS input pull-up function that can be controlled by software. This MOS input pull-up function can be used in modes 1, 2, 4, 5, and 6* when 8-bit bus mode is selected, or in mode 3 or 7*, and can be specified as on or off on an individual bit basis.

When a PEDDR bit is cleared to 0 in mode 1, 2, 4, 5, or 6* when 8-bit bus mode is selected, or in mode 3 or 7*, setting the corresponding PEPCR bit to 1 turns on the MOS input pull-up for that pin.

The MOS input pull-up function is in the off state after a power-on reset, and in hardware standby mode. The prior state is retained after a manual reset, and in software standby mode.

Table 8.18 summarizes the MOS input pull-up states.

**Table 8.18 MOS Input Pull-Up States (Port E)**

| Modes | | Power-On Reset | Hardware Standby Mode | Manual Reset | Software Standby Mode | In Other Operations |
|---|---|---|---|---|---|---|
| 3, 7* | | OFF | | ON/OFF | | |
| 1, 2, 4 to 6* | 8-bit bus | | | | | |
| | 16-bit bus | | | OFF | | |

**Legend:**

OFF : MOS input pull-up is always off.

ON/OFF : On when PEDDR = 0 and PEPCR = 1; otherwise off.

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

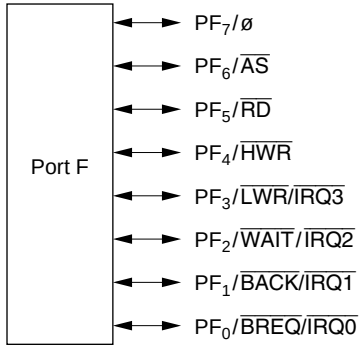
## 8.11 Port F

### 8.11.1 Overview

Port F is an 8-bit I/O port. Port F pins also function as bus control signal input/output pins ($\overline{AS}$, $\overline{RD}$, $\overline{HWR}$, $\overline{LWR}$, $\overline{WAIT}$, $\overline{BREQ}$, and $\overline{BACK}$), the system clock (ø) output pin and interrupt input pins ($\overline{IRQ0}$ to $\overline{IRQ3}$).

The interrupt input pins ($\overline{IRQ0}$ to $\overline{IRQ3}$) are Schmitt-triggered inputs.

Figure 8.23 shows the port F pin configuration.




| Port F pins | Pin functions in modes 1, 2, 4, 5, and 6* |
|---|---|
| $PF_7$/ø | $PF_7$ (input)/ø(output) |
| $PF_6$/$\overline{AS}$ | $\overline{AS}$ (output) |
| $PF_5$/$\overline{RD}$ | $\overline{RD}$ (output) |
| $PF_4$/$\overline{HWR}$ | $\overline{HWR}$ (output) |
| $PF_3$/$\overline{LWR}$/$\overline{IRQ3}$ | $\overline{LWR}$ (output) |
| $PF_2$/$\overline{WAIT}$/$\overline{IRQ2}$ | $PF_2$ (I/O)/$\overline{WAIT}$ (input)/$\overline{IRQ2}$ (input) |
| $PF_1$/$\overline{BACK}$/$\overline{IRQ1}$ | $PF_1$ (I/O)/$\overline{BACK}$ (output)/$\overline{IRQ1}$ (input) |
| $PF_0$/$\overline{BREQ}$/$\overline{IRQ0}$ | $PF_0$ (I/O)/$\overline{BREQ}$ (input)/$\overline{IRQ0}$ (input) |
| | **Pin functions in modes 3 and 7*** |
| | $PF_7$ (input)/ø (output) |
| | $PF_6$ (I/O) |
| | $PF_5$ (I/O) |
| | $PF_4$ (I/O) |
| | $PF_3$ (I/O)/$\overline{IRQ3}$ (input) |
| | $PF_2$ (I/O)/$\overline{IRQ2}$ (input) |
| | $PF_1$ (I/O)/$\overline{IRQ1}$ (input) |
| | $PF_0$ (I/O)/$\overline{IRQ0}$ (input) |

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

**Figure 8.23 Port F Pin Functions**

### 8.11.2 Register Configuration

Table 8.19 shows the port F register configuration.

**Table 8.19 Port F Registers**

| Name | Abbreviation | R/W | Initial Value | Address *[1] |
|---|---|---|---|---|
| Port F data direction register | PFDDR | W | H'80/H'00*[2] | H'FEBE |
| Port F data register | PFDR | R/W | H'00 | H'FF6E |
| Port F register | PORTF | R | Undefined | H'FF5E |

Notes: 1. Lower 16 bits of the address.
2. Initial value depends on the mode.

**Port F Data Direction Register (PFDDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PF7DDR | PF6DDR | PF5DDR | PF4DDR | PF3DDR | PF2DDR | PF1DDR | PF0DDR |
| Modes 1, 2, 4, 5, 6* | | | | | | | | | |
| Initial value | : | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | W | W | W | W | W | W | W | W |
| Modes 3 and 7* | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | W | W | W | W | W | W | W | W |

PFDDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port F. PFDDR cannot be read; if it is, an undefined value will be read.

PFDDR is initialized by a power-on reset, and in hardware standby mode, to H'80 in modes 1, 2, 4, 5, and 6*, and to H'00 in modes 3 and 7*. It retains its prior state after a manual reset, and in software standby mode. The OPE bit in SBYCR is used to select whether the bus control output pins retain their output state or become high-impedance when a transition is made to software standby mode.

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

- Modes 1, 2, 4, 5, and 6*

  Pin $PF_7$ functions as the ø output pin when the corresponding PFDDR bit is set to 1, and as an input port when the bit is cleared to 0.

  The input/output direction specified by PFDDR is ignored for pins $PF_6$ to $PF_3$, which are automatically designated as bus control outputs ($\overline{AS}$, $\overline{RD}$, $\overline{HWR}$, and $\overline{LWR}$).

  Pins $PF_2$ to $PF_0$ are designated as bus control input/output pins ($\overline{WAIT}$, $\overline{BACK}$, $\overline{BREQ}$) by means of bus controller settings. At other times, setting a PFDDR bit to 1 makes the corresponding port F pin an output port, while clearing the bit to 0 makes the pin an input port.

- Modes 3 and 7*

  Setting a PFDDR bit to 1 makes the corresponding port F pin $PF_6$ to $PF_0$ an output port, or in the case of pin $PF_7$, the ø output pin. Clearing the bit to 0 makes the pin an input port.

**Port F Data Register (PFDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PF7DR | PF6DR | PF5DR | PF4DR | PF3DR | PF2DR | PF1DR | PF0DR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

PFDR is an 8-bit readable/writable register that stores output data for the port F pins ($PF_7$ to $PF_0$).

PFDR is initialized to H'00 by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

**Port F Register (PORTF)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PF7 | PF6 | PF5 | PF4 | PF3 | PF2 | PF1 | PF0 |
| Initial value | : | —* | —* | —* | —* | —* | —* | —* | —* |
| R/W | : | R | R | R | R | R | R | R | R |

Note: * Determined by state of pins $PF_7$ to $PF_0$.

PORTF is an 8-bit read-only register that shows the pin states. Writing of output data for the port F pins ($PF_7$ to $PF_0$) must always be performed on PFDR.

If a port F read is performed while PFDDR bits are set to 1, the PFDR values are read. If a port F read is performed while PFDDR bits are cleared to 0, the pin states are read.

After a power-on reset and in hardware standby mode, PORTF contents are determined by the pin states, as PFDDR and PFDR are initialized. PORTF retains its prior state after a manual reset, and in software standby mode.

### 8.11.3 Pin Functions

Port F pins also function as bus control signal input/output pins ($\overline{AS}$, $\overline{RD}$, $\overline{HWR}$, $\overline{LWR}$, $\overline{WAIT}$, $\overline{BREQ}$, and $\overline{BACK}$) the system clock (ø) output pin and interrupt input pins ($\overline{IRQ0}$ to $\overline{IRQ3}$). The pin functions differ between modes 1, 2, 4, 5, and 6*, and modes 3 and 7*. Port F pin functions are shown in table 8.20.

**Table 8.20 Port F Pin Functions**

**Pin** | **Selection Method and Pin Functions**

$PF_7$/ø

The pin function is switched as shown below according to bit PF7DDR.

| PF7DDR | 0 | 1 |
|---|---|---|
| Pin function | $PF_7$ input pin | ø output pin |

$PF_6$/$\overline{AS}$

The pin function is switched as shown below according to the operating mode and bit PF6DDR.

| Operating Mode | Modes 1, 2, 4, 5, 6* | Modes 3 and 7* | |
|---|---|---|---|
| PF6DDR | — | 0 | 1 |
| Pin function | $\overline{AS}$ output pin | $PF_6$ input pin | $PF_6$ output pin |

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

$PF_5$/$\overline{RD}$

The pin function is switched as shown below according to the operating mode and bit PF5DDR.

| Operating Mode | Modes 1, 2, 4, 5, 6* | Modes 3 and 7* | |
|---|---|---|---|
| PF5DDR | — | 0 | 1 |
| Pin function | $\overline{RD}$ output pin | $PF_5$ input pin | $PF_5$ output pin |

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

$PF_4$/$\overline{HWR}$

The pin function is switched as shown below according to the operating mode and bit PF4DDR.

| Operating Mode | Modes 1, 2, 4, 5, 6* | Modes 3 and 7* | |
|---|---|---|---|
| PF4DDR | — | 0 | 1 |
| Pin function | $\overline{HWR}$ output pin | $PF_4$ input pin | $PF_4$ output pin |

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

**Table 8.20 Port F Pin Functions (cont)**

**Pin** | **Selection Method and Pin Functions**

$PF_3/\overline{LWR}/\overline{IRQ3}$ The pin function is switched as shown below according to the operating mode and bit PF3DDR.

| Operating Mode | Modes 1, 2, 4, 5, 6*1 | Modes 3 and 7*1 | |
|---|---|---|---|
| PF3DDR | — | 0 | 1 |
| Pin function | $\overline{LWR}$ output pin | $PF_3$ input pin | $PF_3$ output pin |
| | $\overline{IRQ3}$ interrupt input pin*2 | | |

Notes: 1. Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.
2. When this pin is used as an external interrupt input, the pin function should be set as a port ($PF_3$) input pin.

$PF_2/\overline{WAIT}/\overline{IRQ2}$ The pin function is switched as shown below according to the operating mode, and WAITE bit in BCRL, and PF2DDR bit.

| Operating Mode | Modes 1, 2, 4, 5, 6*1 | | | Modes 3 and 7*1 | |
|---|---|---|---|---|---|
| WAITE | 0 | | 1 | — | |
| PF2DDR | 0 | 1 | — | 0 | 1 |
| Pin function | $PF_2$ input pin | $PF_2$ output pin | $\overline{WAIT}$ input pin | $PF_2$ input pin | $PF_2$ output pin |
| | $\overline{IRQ2}$ interrupt input pin*2 | | | | |

Notes: 1. Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.
2. When this pin is used as an external interrupt input, the pin function should be set as a port ($PF_2$) input pin.

$PF_1/\overline{BACK}/\overline{IRQ1}$ The pin function is switched as shown below according to the operating mode, and the BRLE bit in BCRL and PF1DDR bit.

| Operating Mode | Modes 1, 2, 4, 5, 6*1 | | | Modes 3 and 7*1 | |
|---|---|---|---|---|---|
| BRLE | 0 | | 1 | — | |
| PF1DDR | 0 | 1 | — | 0 | 1 |
| Pin function | $PF_1$ input pin | $PF_1$ output pin | $\overline{BACK}$ output pin | $PF_1$ input pin | $PF_1$ output pin |
| | $\overline{IRQ1}$ interrupt input pin*2 | | | | |

Notes: 1. Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.
2. When this pin is used as an external interrupt input, the pin function should be set as a port ($PF_1$) input pin.

**Table 8.20 Port F Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $PF_0$/$\overline{BREQ}$/$\overline{IRQ0}$ | The pin function is switched as shown below according to the operating mode, and the BRLE bit in BCRL and PF0DDR bit. |

| Operating Mode | Modes 1, 2, 4, 5, 6*1 | | | Modes 3 and 7*1 | |
|---|---|---|---|---|---|
| BRLE | 0 | | 1 | — | |
| PF0DDR | 0 | 1 | — | 0 | 1 |
| Pin function | $PF_0$ input pin | $PF_0$ output pin | $\overline{BREQ}$ input pin | $PF_0$ input pin | $PF_0$ output pin |
| | $\overline{IRQ0}$ interrupt input pin*2 | | | | |

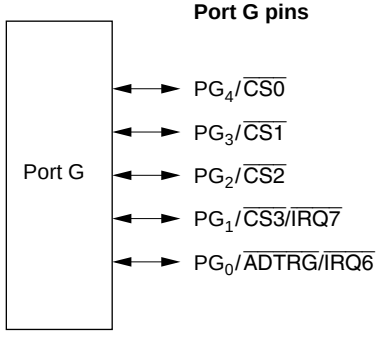
Notes: 1. Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.
2. When this pin is used as an external interrupt input, the pin function should be set as a port ($PF_0$) input pin.

## 8.12 Port G

### 8.12.1 Overview

Port G is a 5-bit I/O port. Port G pins also function as bus control signal output pins ($\overline{CS0}$ to $\overline{CS3}$). The A/D converter input pin ($\overline{ADTRG}$), and interrupt input pins ($\overline{IRQ6}$, $\overline{IRQ7}$). The interrupt input pins ($\overline{IRQ6}$, $\overline{IRQ7}$) are Schmitt-triggered inputs.

Figure 8.24 shows the port G pin configuration.




**Pin functions in modes 1 and 2***

$PG_4$ (input)/$\overline{CS0}$ (output)
$PG_3$ (I/O)
$PG_2$ (I/O)
$PG_1$ (I/O)/$\overline{IRQ7}$ (input)
$PG_0$ (I/O)/$\overline{ADTRG}$ (input)/$\overline{IRQ6}$ (input)

**Pin functions in modes 3 and 7***

$PG_4$ (I/O)
$PG_3$ (I/O)
$PG_2$ (I/O)
$PG_1$ (I/O)/$\overline{IRQ7}$ (input)
$PG_0$ (I/O)/$\overline{ADTRG}$ (input)/$\overline{IRQ6}$ (input)

**Pin functions in modes 4 to 6***

$PG_4$ (input)/$\overline{CS0}$ (output)
$PG_3$ (input)/$\overline{CS1}$ (output)
$PG_2$ (input)/$\overline{CS2}$ (output)
$PG_1$ (input)/$\overline{CS3}$ (output)/$\overline{IRQ7}$ (input)
$PG_0$ (I/O)/$\overline{ADTRG}$ (input)/$\overline{IRQ6}$ (input)

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

**Figure 8.24 Port G Pin Functions**

### 8.12.2 Register Configuration

Table 8.21 shows the port G register configuration.

**Table 8.21 Port G Registers**

| Name | Abbreviation | R/W | Initial Value*[1] | Address*[2] |
|---|---|---|---|---|
| Port G data direction register | PGDDR | W | H'10/H'00*[3] | H'FEBF |
| Port G data register | PGDR | R/W | H'00 | H'FF6F |
| Port G register | PORTG | R | Undefined | H'FF5F |

Notes: 1. Value of bits 4 to 0.
2. Lower 16 bits of the address.
3. Initial value depends on the mode.

**Port G Data Direction Register (PGDDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | PG4DDR | PG3DDR | PG2DDR | PG1DDR | PG0DDR |
| Modes 1, 4, 5* | | | | | | | | | |
| Initial value | : | Undefined | Undefined | Undefined | 1 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | — | W | W | W | W | W |
| Modes 2, 3, 6, 7* | | | | | | | | | |
| Initial value | : | Undefined | Undefined | Undefined | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | — | W | W | W | W | W |

PGDDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port G. PGDDR cannot be read, and bits 7 to 5 are reserved. If PGDDR is read, an undefined value will be read.

The PGDDR is initialized by a power-on reset and in hardware standby mode, to H'10 (bits 4 to 0) in modes 1, 4, and 5*, and to H'00 (bits 4 to 0) in modes 2, 3, 6, and 7*. It retains its prior state after a manual reset and in software standby mode. The OPE bit in SBYCR is used to select whether the bus control output pins retain their output state or become high-impedance when a transition is made to software standby mode.

- Modes 1 and 2*

  Pin $PG_4$ functions as a bus control output pin ($\overline{CS0}$) when the corresponding PGDDR bit is set to 1, and as an input port when the bit is cleared to 0.

  For pins $PG_3$ to $PG_0$, setting the corresponding PGDDR bit to 1 makes the pin an output port, while clearing the bit to 0 makes the pin an input port.

- Modes 3 and 7*

  Setting a PGDDR bit to 1 makes the corresponding port G pin an output port, while clearing the bit to 0 makes the pin an input port.

- Modes 4, 5, and 6*

  Pins $PG_4$ to $PG_1$ function as bus control output pins ($\overline{CS0}$ to $\overline{CS3}$) when the corresponding PGDDR bits are set to 1, and as input ports when the bits are cleared to 0.

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

**Port G Data Register (PGDR)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | PG4DR | PG3DR | PG2DR | PG1DR | PG0DR |
| Initial value | : | Undefined | Undefined | Undefined | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | — | R/W | R/W | R/W | R/W | R/W |

PGDR is an 8-bit readable/writable register that stores output data for the port G pins ($PG_4$ to $PG_0$).

Bits 7 to 5 are reserved; they return an undetermined value if read, and cannot be modified.

PGDR is initialized to H'00 (bits 4 to 0) by a power-on reset, and in hardware standby mode. It retains its prior state after a manual reset, and in software standby mode.

**Port G Register (PORTG)**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | PG4 | PG3 | PG2 | PG1 | PG0 |
| Initial value | : | Undefined | Undefined | Undefined | —* | —* | —* | —* | —* |
| R/W | : | — | — | — | R | R | R | R | R |

Note: * Determined by state of pins $PG_4$ to $PG_0$.

PORTG is an 8-bit read-only register that shows the pin states. It cannot be written to. Writing of output data for the port G pins ($PG_4$ to $PG_0$) must always be performed on PGDR.

Bits 7 to 5 are reserved; they return an undetermined value if read, and cannot be modified.

If a port G read is performed while PGDDR bits are set to 1, the PGDR values are read. If a port G read is performed while PGDDR bits are cleared to 0, the pin states are read.

After a power-on reset and in hardware standby mode, PORTG contents are determined by the pin states, as PGDDR and PGDR are initialized. PORTG retains its prior state after a manual reset, and in software standby mode.

### 8.12.3 Pin Functions

Port G pins also function as bus control signal output pins ($\overline{CS0}$ to $\overline{CS3}$) the A/D converter input pin ($\overline{ADTRG}$), and interrupt input pins ($\overline{IRQ6}$, $\overline{IRQ7}$). The pin functions are different in modes 1 and 2, modes 3 and 7, and modes 4 to 6. Port G pin functions are shown in table 8.22.

**Table 8.22 Port G Pin Functions**

| Pin | Selection Method and Pin Functions |
|---|---|
| $PG_4/\overline{CS0}$ | The pin function is switched as shown below according to the operating mode and bit PG4DDR. |

| Operating Mode | Modes 1, 2, 4, 5, 6* | | Modes 3 and 7* | |
|---|---|---|---|---|
| PG4DDR | 0 | 1 | 0 | 1 |
| Pin function | $PG_4$ input pin | $\overline{CS0}$ output pin | $PG_4$ input pin | $PG_4$ output pin |

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

$PG_3/\overline{CS1}$ — The pin function is switched as shown below according to the operating mode and bit PG3DDR.

| Operating Mode | Modes 1, 2, 3, 7* | | Modes 4 to 6* | |
|---|---|---|---|---|
| PG3DDR | 0 | 1 | 0 | 1 |
| Pin function | $PG_3$ input pin | $PG_3$ output pin | $PG_3$ input pin | $\overline{CS1}$ output pin |

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

$PG_2/\overline{CS2}$ — The pin function is switched as shown below according to the operating mode and bit PG2DDR.

| Operating Mode | Modes 1, 2, 3, 7* | | Modes 4 to 6* | |
|---|---|---|---|---|
| PG2DDR | 0 | 1 | 0 | 1 |
| Pin function | $PG_2$ input pin | $PG_2$ output pin | $PG_2$ input pin | $\overline{CS2}$ output pin |

Note: * Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.

**Table 8.22 Port G Pin Functions (cont)**

| Pin | Selection Method and Pin Functions |
|---|---|
| $PG_1$/$\overline{CS3}$/$\overline{IRQ7}$ | The pin function is switched as shown below according to the combination of operating mode and bit PG1DDR. |

| Operating Mode | Modes 1, 2, 3, 7*[1] | | Modes 4 to 6*[1] | |
|---|---|---|---|---|
| PG1DDR | 0 | 1 | 0 | 1 |
| Pin function | $PG_1$ input pin | $PG_1$ output pin | $PG_1$ input pin | $\overline{CS3}$ output pin |
| | $\overline{IRQ7}$ interrupt input pin*[2] | | | |

Notes: 1. Modes 1 to 3 are not available on the F-ZTAT version.
Modes 2, 3, 6, and 7 are not available on the ROMless version.
2. When this pin is used as an external interrupt input, it should not be used as an input/output pin with other functions.

$PG_0$/$\overline{ADTRG}$/$\overline{IRQ6}$ The pin function is switched as shown below according to the combination of bits TRGS1 and TRGS0 (trigger select 1 and 0) in the A/D control register (ADCR).

| PG0DDR | 0 | 1 |
|---|---|---|
| Pin function | $PG_0$ input | $PG_0$ output |
| | $\overline{ADTRG}$ input pin*[1] | |
| | $\overline{IRQ6}$ interrupt input pin*[2] | |

Notes: 1. $\overline{ADTRG}$ input when TRGS1 = TRGS0 = 1.
2. When this pin is used as an external interrupt input, it should not be used as an input/output pin with other functions.

# Section 9 16-Bit Timer Pulse Unit (TPU)

## 9.1 Overview

The H8S/2345 Series has an on-chip 16-bit timer pulse unit (TPU) that comprises six 16-bit timer channels.

### 9.1.1 Features

- Maximum 16-pulse input/output
  - A total of 16 timer general registers (TGRs) are provided (four each for channels 0 and 3, and two each for channels 1, 2, 4, and 5), each of which can be set independently as an output compare/input capture register
  - TGRC and TGRD for channels 0 and 3 can also be used as buffer registers
- Selection of 8 counter input clocks for each channel
- The following operations can be set for each channel:
  - Waveform output at compare match: Selection of 0, 1, or toggle output
  - Input capture function: Selection of rising edge, falling edge, or both edge detection
  - Counter clear operation: Counter clearing possible by compare match or input capture
  - Synchronous operation: Multiple timer counters (TCNT) can be written to simultaneously
    Simultaneous clearing by compare match and input capture possible
    Register simultaneous input/output possible by counter synchronous operation
  - PWM mode: Any PWM output duty can be set
    Maximum of 15-phase PWM output possible by combination with synchronous operation
- Buffer operation settable for channels 0 and 3
  - Input capture register double-buffering possible
  - Automatic rewriting of output compare register possible
- Phase counting mode settable independently for each of channels 1, 2, 4, and 5
  - Two-phase encoder pulse up/down-count possible
- Cascaded operation
  - Channel 2 (channel 5) input clock operates as 32-bit counter by setting channel 1 (channel 4) overflow/underflow
- Fast access via internal 16-bit bus
  - Fast access is possible via a 16-bit bus interface

- 26 interrupt sources
  - For channels 0 and 3, four compare match/input capture dual-function interrupts and one overflow interrupt can be requested independently
  - For channels 1, 2, 4, and 5, two compare match/input capture dual-function interrupts, one overflow interrupt, and one underflow interrupt can be requested independently
- Automatic transfer of register data
  - Block transfer, 1-word data transfer, and 1-byte data transfer possible by data transfer controller (DTC) activation
- A/D converter conversion start trigger can be generated
  - Channel 0 to 5 compare match A/input capture A signals can be used as A/D converter conversion start trigger
- Module stop mode can be set
  - As the initial setting, TPU operation is halted. Register access is enabled by exiting module stop mode.

Table 9.1 lists the functions of the TPU.

**Table 9.1 TPU Functions**

| Item | | Channel 0 | Channel 1 | Channel 2 | Channel 3 | Channel 4 | Channel 5 |
|---|---|---|---|---|---|---|---|
| Count clock | | ø/1<br>ø/4<br>ø/16<br>ø/64<br>TCLKA<br>TCLKB<br>TCLKC<br>TCLKD | ø/1<br>ø/4<br>ø/16<br>ø/64<br>ø/256<br>TCLKA<br>TCLKB | ø/1<br>ø/4<br>ø/16<br>ø/64<br>ø/1024<br>TCLKA<br>TCLKB<br>TCLKC | ø/1<br>ø/4<br>ø/16<br>ø/64<br>ø/256<br>ø/1024<br>ø/4096<br>TCLKA | ø/1<br>ø/4<br>ø/16<br>ø/64<br>ø/1024<br>TCLKA<br>TCLKC | ø/1<br>ø/4<br>ø/16<br>ø/64<br>ø/256<br>TCLKA<br>TCLKC<br>TCLKD |
| General registers | | TGR0A<br>TGR0B | TGR1A<br>TGR1B | TGR2A<br>TGR2B | TGR3A<br>TGR3B | TGR4A<br>TGR4B | TGR5A<br>TGR5B |
| General registers/ buffer registers | | TGR0C<br>TGR0D | — | — | TGR3C<br>TGR3D | — | — |
| I/O pins | | TIOCA0<br>TIOCB0<br>TIOCC0<br>TIOCD0 | TIOCA1<br>TIOCB1 | TIOCA2<br>TIOCB2 | TIOCA3<br>TIOCB3<br>TIOCC3<br>TIOCD3 | TIOCA4<br>TIOCB4 | TIOCA5<br>TIOCB5 |
| Counter clear function | | TGR compare match or input capture | TGR compare match or input capture | TGR compare match or input capture | TGR compare match or input capture | TGR compare match or input capture | TGR compare match or input capture |
| Compare match output | 0 output | ○ | ○ | ○ | ○ | ○ | ○ |
| | 1 output | ○ | ○ | ○ | ○ | ○ | ○ |
| | Toggle output | ○ | ○ | ○ | ○ | ○ | ○ |
| Input capture function | | ○ | ○ | ○ | ○ | ○ | ○ |
| Synchronous operation | | ○ | ○ | ○ | ○ | ○ | ○ |
| PWM mode | | ○ | ○ | ○ | ○ | ○ | ○ |
| Phase counting mode | | — | ○ | ○ | — | ○ | ○ |
| Buffer operation | | ○ | — | — | ○ | — | — |
| DTC activation | | TGR compare match or input capture | TGR compare match or input capture | TGR compare match or input capture | TGR compare match or input capture | TGR compare match or input capture | TGR compare match or input capture |

**Table 9.1 TPU Functions (cont)**

| Item | Channel 0 | Channel 1 | Channel 2 | Channel 3 | Channel 4 | Channel 5 |
|---|---|---|---|---|---|---|
| A/D converter trigger | TGR0A compare match or input capture | TGR1A compare match or input capture | TGR2A compare match or input capture | TGR3A compare match or input capture | TGR4A compare match or input capture | TGR5A compare match or input capture |
| Interrupt sources | 5 sources<br>• Compare match or input capture 0A<br>• Compare match or input capture 0B<br>• Compare match or input capture 0C<br>• Compare match or input capture 0D<br>• Overflow | 4 sources<br>• Compare match or input capture 1A<br>• Compare match or input capture 1B<br>• Overflow<br>• Underflow | 4 sources<br>• Compare match or input capture 2A<br>• Compare match or input capture 2B<br>• Overflow<br>• Underflow | 5 sources<br>• Compare match or input capture 3A<br>• Compare match or input capture 3B<br>• Compare match or input capture 3C<br>• Compare match or input capture 3D<br>• Overflow | 4 sources<br>• Compare match or input capture 4A<br>• Compare match or input capture 4B<br>• Overflow<br>• Underflow | 4 sources<br>• Compare match or input capture 5A<br>• Compare match or input capture 5B<br>• Overflow<br>• Underflow |

Legend

—: Not possible

○: Possible

### 9.1.2 Block Diagram

Figure 9.1 shows a block diagram of the TPU.

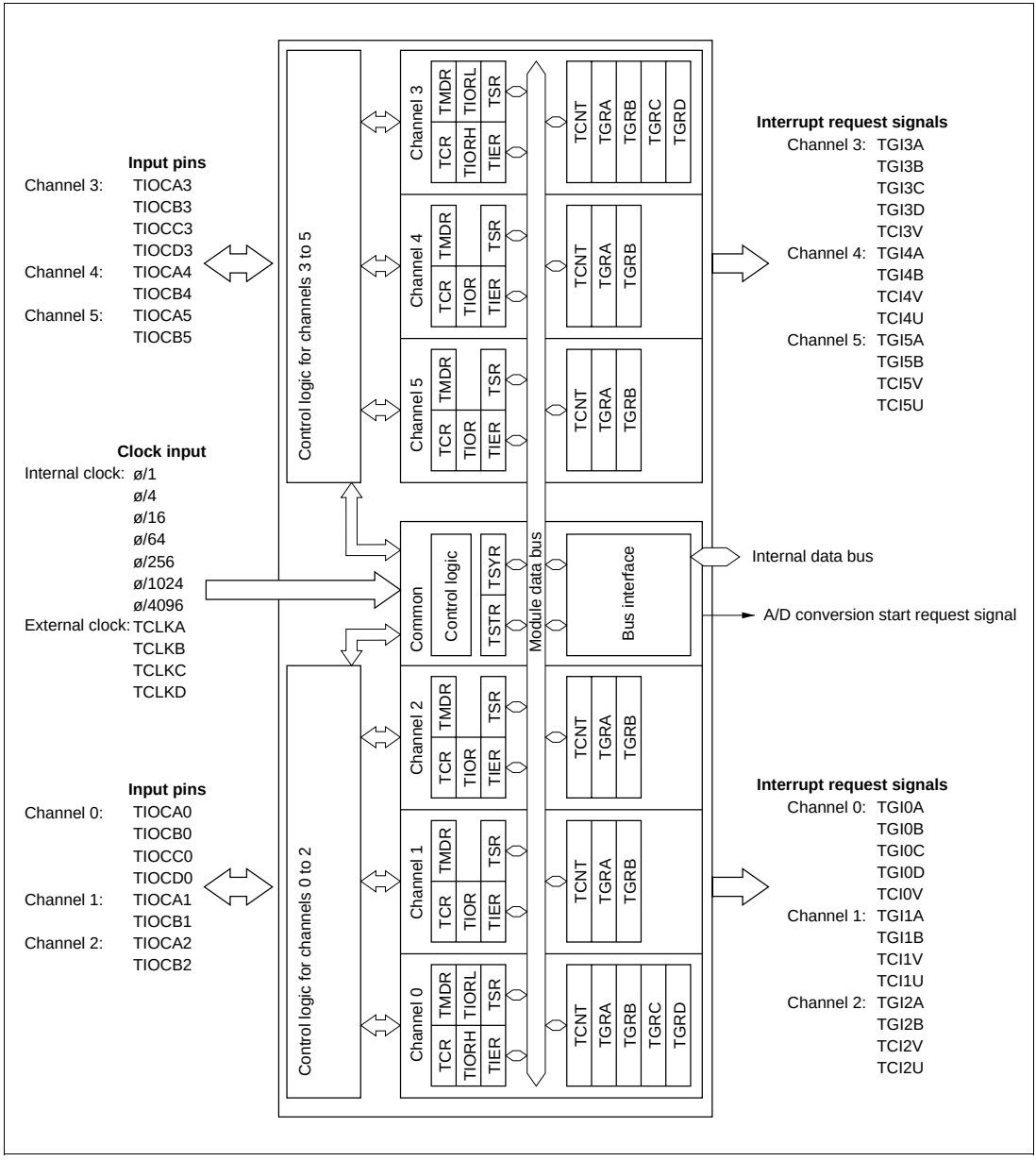



**Figure 9.1 Block Diagram of TPU**

### 9.1.3 Pin Configuration

Table 9.2 summarizes the TPU pins.

**Table 9.2 TPU Pins**

| Channel | Name | Symbol | I/O | Function |
|---|---|---|---|---|
| All | Clock input A | TCLKA | Input | External clock A input pin (Channel 1 and 5 phase counting mode A phase input) |
| | Clock input B | TCLKB | Input | External clock B input pin (Channel 1 and 5 phase counting mode B phase input) |
| | Clock input C | TCLKC | Input | External clock C input pin (Channel 2 and 4 phase counting mode A phase input) |
| | Clock input D | TCLKD | Input | External clock D input pin (Channel 2 and 4 phase counting mode B phase input) |
| 0 | Input capture/out compare match A0 | TIOCA0 | I/O | TGR0A input capture input/output compare output/PWM output pin |
| | Input capture/out compare match B0 | TIOCB0 | I/O | TGR0B input capture input/output compare output/PWM output pin |
| | Input capture/out compare match C0 | TIOCC0 | I/O | TGR0C input capture input/output compare output/PWM output pin |
| | Input capture/out compare match D0 | TIOCD0 | I/O | TGR0D input capture input/output compare output/PWM output pin |
| 1 | Input capture/out compare match A1 | TIOCA1 | I/O | TGR1A input capture input/output compare output/PWM output pin |
| | Input capture/out compare match B1 | TIOCB1 | I/O | TGR1B input capture input/output compare output/PWM output pin |
| 2 | Input capture/out compare match A2 | TIOCA2 | I/O | TGR2A input capture input/output compare output/PWM output pin |
| | Input capture/out compare match B2 | TIOCB2 | I/O | TGR2B input capture input/output compare output/PWM output pin |

**Table 9.2 TPU Pins (cont)**

| Channel | Name | Symbol | I/O | Function |
|---|---|---|---|---|
| 3 | Input capture/out compare match A3 | TIOCA3 | I/O | TGR3A input capture input/output compare output/PWM output pin |
| | Input capture/out compare match B3 | TIOCB3 | I/O | TGR3B input capture input/output compare output/PWM output pin |
| | Input capture/out compare match C3 | TIOCC3 | I/O | TGR3C input capture input/output compare output/PWM output pin |
| | Input capture/out compare match D3 | TIOCD3 | I/O | TGR3D input capture input/output compare output/PWM output pin |
| 4 | Input capture/out compare match A4 | TIOCA4 | I/O | TGR4A input capture input/output compare output/PWM output pin |
| | Input capture/out compare match B4 | TIOCB4 | I/O | TGR4B input capture input/output compare output/PWM output pin |
| 5 | Input capture/out compare match A5 | TIOCA5 | I/O | TGR5A input capture input/output compare output/PWM output pin |
| | Input capture/out compare match B5 | TIOCB5 | I/O | TGR5B input capture input/output compare output/PWM output pin |

### 9.1.4 Register Configuration

Table 9.3 summarizes the TPU registers.

**Table 9.3 TPU Registers**

| Channel | Name | Abbreviation | R/W | Initial Value | Address *[1] |
|---|---|---|---|---|---|
| 0 | Timer control register 0 | TCR0 | R/W | H'00 | H'FFD0 |
| | Timer mode register 0 | TMDR0 | R/W | H'C0 | H'FFD1 |
| | Timer I/O control register 0H | TIOR0H | R/W | H'00 | H'FFD2 |
| | Timer I/O control register 0L | TIOR0L | R/W | H'00 | H'FFD3 |
| | Timer interrupt enable register 0 | TIER0 | R/W | H'40 | H'FFD4 |
| | Timer status register 0 | TSR0 | R/(W)*[2] | H'C0 | H'FFD5 |
| | Timer counter 0 | TCNT0 | R/W | H'0000 | H'FFD6 |
| | Timer general register 0A | TGR0A | R/W | H'FFFF | H'FFD8 |
| | Timer general register 0B | TGR0B | R/W | H'FFFF | H'FFDA |
| | Timer general register 0C | TGR0C | R/W | H'FFFF | H'FFDC |
| | Timer general register 0D | TGR0D | R/W | H'FFFF | H'FFDE |
| 1 | Timer control register 1 | TCR1 | R/W | H'00 | H'FFE0 |
| | Timer mode register 1 | TMDR1 | R/W | H'C0 | H'FFE1 |
| | Timer I/O control register 1 | TIOR1 | R/W | H'00 | H'FFE2 |
| | Timer interrupt enable register 1 | TIER1 | R/W | H'40 | H'FFE4 |
| | Timer status register 1 | TSR1 | R/(W) *[2] | H'C0 | H'FFE5 |
| | Timer counter 1 | TCNT1 | R/W | H'0000 | H'FFE6 |
| | Timer general register 1A | TGR1A | R/W | H'FFFF | H'FFE8 |
| | Timer general register 1B | TGR1B | R/W | H'FFFF | H'FFEA |
| 2 | Timer control register 2 | TCR2 | R/W | H'00 | H'FFF0 |
| | Timer mode register 2 | TMDR2 | R/W | H'C0 | H'FFF1 |
| | Timer I/O control register 2 | TIOR2 | R/W | H'00 | H'FFF2 |
| | Timer interrupt enable register 2 | TIER2 | R/W | H'40 | H'FFF4 |
| | Timer status register 2 | TSR2 | R/(W) *[2] | H'C0 | H'FFF5 |
| | Timer counter 2 | TCNT2 | R/W | H'0000 | H'FFF6 |
| | Timer general register 2A | TGR2A | R/W | H'FFFF | H'FFF8 |
| | Timer general register 2B | TGR2B | R/W | H'FFFF | H'FFFA |

**Table 9.3 TPU Registers (cont)**

| Channel | Name | Abbreviation | R/W | Initial Value | Address*[1] |
|---|---|---|---|---|---|
| 3 | Timer control register 3 | TCR3 | R/W | H'00 | H'FE80 |
| | Timer mode register 3 | TMDR3 | R/W | H'C0 | H'FE81 |
| | Timer I/O control register 3H | TIOR3H | R/W | H'00 | H'FE82 |
| | Timer I/O control register 3L | TIOR3L | R/W | H'00 | H'FE83 |
| | Timer interrupt enable register 3 | TIER3 | R/W | H'40 | H'FE84 |
| | Timer status register 3 | TSR3 | R/(W)*[2] | H'C0 | H'FE85 |
| | Timer counter 3 | TCNT3 | R/W | H'0000 | H'FE86 |
| | Timer general register 3A | TGR3A | R/W | H'FFFF | H'FE88 |
| | Timer general register 3B | TGR3B | R/W | H'FFFF | H'FE8A |
| | Timer general register 3C | TGR3C | R/W | H'FFFF | H'FE8C |
| | Timer general register 3D | TGR3D | R/W | H'FFFF | H'FE8E |
| 4 | Timer control register 4 | TCR4 | R/W | H'00 | H'FE90 |
| | Timer mode register 4 | TMDR4 | R/W | H'C0 | H'FE91 |
| | Timer I/O control register 4 | TIOR4 | R/W | H'00 | H'FE92 |
| | Timer interrupt enable register 4 | TIER4 | R/W | H'40 | H'FE94 |
| | Timer status register 4 | TSR4 | R/(W) *[2] | H'C0 | H'FE95 |
| | Timer counter 4 | TCNT4 | R/W | H'0000 | H'FE96 |
| | Timer general register 4A | TGR4A | R/W | H'FFFF | H'FE98 |
| | Timer general register 4B | TGR4B | R/W | H'FFFF | H'FE9A |
| 5 | Timer control register 5 | TCR5 | R/W | H'00 | H'FEA0 |
| | Timer mode register 5 | TMDR5 | R/W | H'C0 | H'FEA1 |
| | Timer I/O control register 5 | TIOR5 | R/W | H'00 | H'FEA2 |
| | Timer interrupt enable register 5 | TIER5 | R/W | H'40 | H'FEA4 |
| | Timer status register 5 | TSR5 | R/(W) *[2] | H'C0 | H'FEA5 |
| | Timer counter 5 | TCNT5 | R/W | H'0000 | H'FEA6 |
| | Timer general register 5A | TGR5A | R/W | H'FFFF | H'FEA8 |
| | Timer general register 5B | TGR5B | R/W | H'FFFF | H'FEAA |
| All | Timer start register | TSTR | R/W | H'00 | H'FFC0 |
| | Timer synchro register | TSYR | R/W | H'00 | H'FFC1 |
| | Module stop control register | MSTPCR | R/W | H'3FFF | H'FF3C |

Notes: 1. Lower 16 bits of the address.
2. Can only be written with 0 for flag clearing.

## 9.2 Register Descriptions

### 9.2.1 Timer Control Register (TCR)

**Channel 0: TCR0**

**Channel 3: TCR3**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | CCLR2 | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Channel 1: TCR1**

**Channel 2: TCR2**

**Channel 4: TCR4**

**Channel 5: TCR5**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The TCR registers are 8-bit registers that control the TCNT channels. The TPU has six TCR registers, one for each of channels 0 to 5. The TCR registers are initialized to H'00 by a reset, and in hardware standby mode.

Note: Make TCR settings only when TCNT operation is stopped.

**Bits 7, 6, 5—Counter Clear 2, 1, and 0 (CCLR2, CCLR1, CCLR0):** These bits select the TCNT counter clearing source.

| Channel | Bit 7 CCLR2 | Bit 6 CCLR1 | Bit 5 CCLR0 | Description |
|---|---|---|---|---|
| 0, 3 | 0 | 0 | 0 | TCNT clearing disabled (Initial value) |
| | | | 1 | TCNT cleared by TGRA compare match/input capture |
| | | 1 | 0 | TCNT cleared by TGRB compare match/input capture |
| | | | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/ synchronous operation *1 |
| | 1 | 0 | 0 | TCNT clearing disabled |
| | | | 1 | TCNT cleared by TGRC compare match/input capture *2 |
| | | 1 | 0 | TCNT cleared by TGRD compare match/input capture *2 |
| | | | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/ synchronous operation *1 |

| Channel | Bit 7 Reserved*3 | Bit 6 CCLR1 | Bit 5 CCLR0 | Description |
|---|---|---|---|---|
| 1, 2, 4, 5 | 0 | 0 | 0 | TCNT clearing disabled (Initial value) |
| | | | 1 | TCNT cleared by TGRA compare match/input capture |
| | | 1 | 0 | TCNT cleared by TGRB compare match/input capture |
| | | | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/ synchronous operation *1 |

Notes: 1. Synchronous operation setting is performed by setting the SYNC bit in TSYR to 1.
2. When TGRC or TGRD is used as a buffer register, TCNT is not cleared because the buffer register setting has priority, and compare match/input capture does not occur.
3. Bit 7 is reserved in channels 1, 2, 4, and 5. It is always read as 0 and cannot be modified.

**Bits 4 and 3—Clock Edge 1 and 0 (CKEG1, CKEG0):** These bits select the input clock edge. When the input clock is counted using both edges, the input clock period is halved (e.g. ø/4 both edges = ø/2 rising edge). If phase counting mode is used on channels 1, 2, 4, and 5, this setting is ignored and the phase counting mode setting has priority.

| Bit 4 | Bit 3 | | |
|---|---|---|---|
| **CKEG1** | **CKEG0** | **Description** | |
| 0 | 0 | Count at rising edge | (Initial value) |
| | 1 | Count at falling edge | |
| 1 | — | Count at both edges | |

Note: Internal clock edge selection is valid when the input clock is ø/4 or slower. This setting is ignored if the input clock is ø/1, or when overflow/underflow of another channel is selected.

**Bits 2, 1, and 0—Time Prescaler 2, 1, and 0 (TPSC2 to TPSC0):** These bits select the TCNT counter clock. The clock source can be selected independently for each channel. Table 9.4 shows the clock sources that can be set for each channel.

**Table 9.4 TPU Clock Sources**

| | Internal Clock | | | | | | | External Clock | | | | Overflow/ Underflow on Another |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Channel** | **ø/1** | **ø/4** | **ø/16** | **ø/64** | **ø/256** | **ø/1024** | **ø/4096** | **TCLKA** | **TCLKB** | **TCLKC** | **TCLKD** | **Channel** |
| 0 | ○ | ○ | ○ | ○ | | | | ○ | ○ | ○ | ○ | |
| 1 | ○ | ○ | ○ | ○ | ○ | | | ○ | ○ | | | ○ |
| 2 | ○ | ○ | ○ | ○ | | ○ | | ○ | ○ | ○ | | |
| 3 | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | | | | |
| 4 | ○ | ○ | ○ | ○ | | ○ | | ○ | | ○ | | ○ |
| 5 | ○ | ○ | ○ | ○ | ○ | | | ○ | | ○ | ○ | |

**Legend**

○ : Setting

Blank : No setting

| Channel | Bit 2 TPSC2 | Bit 1 TPSC1 | Bit 0 TPSC0 | Description | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Internal clock: counts on ø/1 | (Initial value) |
| | | | 1 | Internal clock: counts on ø/4 | |
| | | 1 | 0 | Internal clock: counts on ø/16 | |
| | | | 1 | Internal clock: counts on ø/64 | |
| | 1 | 0 | 0 | External clock: counts on TCLKA pin input | |
| | | | 1 | External clock: counts on TCLKB pin input | |
| | | 1 | 0 | External clock: counts on TCLKC pin input | |
| | | | 1 | External clock: counts on TCLKD pin input | |

| Channel | Bit 2 TPSC2 | Bit 1 TPSC1 | Bit 0 TPSC0 | Description | |
|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | Internal clock: counts on ø/1 | (Initial value) |
| | | | 1 | Internal clock: counts on ø/4 | |
| | | 1 | 0 | Internal clock: counts on ø/16 | |
| | | | 1 | Internal clock: counts on ø/64 | |
| | 1 | 0 | 0 | External clock: counts on TCLKA pin input | |
| | | | 1 | External clock: counts on TCLKB pin input | |
| | | 1 | 0 | Internal clock: counts on ø/256 | |
| | | | 1 | Counts on TCNT2 overflow/underflow | |

Note: This setting is ignored when channel 1 is in phase counting mode.

| Channel | Bit 2 TPSC2 | Bit 1 TPSC1 | Bit 0 TPSC0 | Description | |
|---|---|---|---|---|---|
| 2 | 0 | 0 | 0 | Internal clock: counts on ø/1 | (Initial value) |
| | | | 1 | Internal clock: counts on ø/4 | |
| | | 1 | 0 | Internal clock: counts on ø/16 | |
| | | | 1 | Internal clock: counts on ø/64 | |
| | 1 | 0 | 0 | External clock: counts on TCLKA pin input | |
| | | | 1 | External clock: counts on TCLKB pin input | |
| | | 1 | 0 | External clock: counts on TCLKC pin input | |
| | | | 1 | Internal clock: counts on ø/1024 | |

Note: This setting is ignored when channel 2 is in phase counting mode.

| Channel | Bit 2 TPSC2 | Bit 1 TPSC1 | Bit 0 TPSC0 | Description | |
|---|---|---|---|---|---|
| 3 | 0 | 0 | 0 | Internal clock: counts on ø/1 | (Initial value) |
| | | | 1 | Internal clock: counts on ø/4 | |
| | | 1 | 0 | Internal clock: counts on ø/16 | |
| | | | 1 | Internal clock: counts on ø/64 | |
| | 1 | 0 | 0 | External clock: counts on TCLKA pin input | |
| | | | 1 | Internal clock: counts on ø/1024 | |
| | | 1 | 0 | Internal clock: counts on ø/256 | |
| | | | 1 | Internal clock: counts on ø/4096 | |

| Channel | Bit 2 TPSC2 | Bit 1 TPSC1 | Bit 0 TPSC0 | Description | |
|---|---|---|---|---|---|
| 4 | 0 | 0 | 0 | Internal clock: counts on ø/1 | (Initial value) |
| | | | 1 | Internal clock: counts on ø/4 | |
| | | 1 | 0 | Internal clock: counts on ø/16 | |
| | | | 1 | Internal clock: counts on ø/64 | |
| | 1 | 0 | 0 | External clock: counts on TCLKA pin input | |
| | | | 1 | External clock: counts on TCLKC pin input | |
| | | 1 | 0 | Internal clock: counts on ø/1024 | |
| | | | 1 | Counts on TCNT5 overflow/underflow | |

Note: This setting is ignored when channel 4 is in phase counting mode.

| Channel | Bit 2 TPSC2 | Bit 1 TPSC1 | Bit 0 TPSC0 | Description | |
|---|---|---|---|---|---|
| 5 | 0 | 0 | 0 | Internal clock: counts on ø/1 | (Initial value) |
| | | | 1 | Internal clock: counts on ø/4 | |
| | | 1 | 0 | Internal clock: counts on ø/16 | |
| | | | 1 | Internal clock: counts on ø/64 | |
| | 1 | 0 | 0 | External clock: counts on TCLKA pin input | |
| | | | 1 | External clock: counts on TCLKC pin input | |
| | | 1 | 0 | Internal clock: counts on ø/256 | |
| | | | 1 | External clock: counts on TCLKD pin input | |

Note: This setting is ignored when channel 5 is in phase counting mode.

### 9.2.2 Timer Mode Register (TMDR)

**Channel 0: TMDR0**

**Channel 3: TMDR3**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | BFB | BFA | MD3 | MD2 | MD1 | MD0 |
| Initial value | : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

**Channel 1: TMDR1**

**Channel 2: TMDR2**

**Channel 4: TMDR4**

**Channel 5: TMDR5**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | MD3 | MD2 | MD1 | MD0 |
| Initial value | : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | — | — | R/W | R/W | R/W | R/W |

The TMDR registers are 8-bit readable/writable registers that are used to set the operating mode for each channel. The TPU has six TMDR registers, one for each channel. The TMDR registers are initialized to H'C0 by a reset, and in hardware standby mode.

Note: Make TMDR settings only when TCNT operation is stopped.

**Bits 7 and 6—Reserved:** Read-only bits, always read as 1.

**Bit 5—Buffer Operation B (BFB):** Specifies whether TGRB is to operate in the normal way, or TGRB and TGRD are to be used together for buffer operation. When TGRD is used as a buffer register, TGRD input capture/output compare is not generated.

In channels 1, 2, 4, and 5, which have no TGRD, bit 5 is reserved. It is always read as 0 and cannot be modified.

**Bit 5**

| BFB | Description | |
|---|---|---|
| 0 | TGRB operates normally | (Initial value) |
| 1 | TGRB and TGRD used together for buffer operation | |

**Bit 4—Buffer Operation A (BFA):** Specifies whether TGRA is to operate in the normal way, or TGRA and TGRC are to be used together for buffer operation. When TGRC is used as a buffer register, TGRC input capture/output compare is not generated.

In channels 1, 2, 4, and 5, which have no TGRC, bit 4 is reserved. It is always read as 0 and cannot be modified.

**Bit 4**

| BFA | Description | |
|---|---|---|
| 0 | TGRA operates normally | (Initial value) |
| 1 | TGRA and TGRC used together for buffer operation | |

**Bits 3 to 0—Modes 3 to 0 (MD3 to MD0):** These bits are used to set the timer operating mode.

| Bit 3 | Bit 2 | Bit 1 | Bit 0 | | |
|---|---|---|---|---|---|
| MD3*[1] | MD2*[2] | MD1 | MD0 | Description | |
| 0 | 0 | 0 | 0 | Normal operation | (Initial value) |
| | | | 1 | Reserved | |
| | | 1 | 0 | PWM mode 1 | |
| | | | 1 | PWM mode 2 | |
| | 1 | 0 | 0 | Phase counting mode 1 | |
| | | | 1 | Phase counting mode 2 | |
| | | 1 | 0 | Phase counting mode 3 | |
| | | | 1 | Phase counting mode 4 | |
| 1 | * | * | * | — | |

*: Don't care

Notes: 1. MD3 is a reserved bit. In a write, it should always be written with 0.
2. Phase counting mode cannot be set for channels 0 and 3. In this case, 0 should always be written to MD2.

### 9.2.3 Timer I/O Control Register (TIOR)

**Channel 0: TIOR0H**
**Channel 1: TIOR1**
**Channel 2: TIOR2**
**Channel 3: TIOR3H**
**Channel 4: TIOR4**
**Channel 5: TIOR5**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Channel 0: TIOR0L**
**Channel 3: TIOR3L**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | IOD3 | IOD2 | IOD1 | IOD0 | IOC3 | IOC2 | IOC1 | IOC0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note: When TGRC or TGRD is designated for buffer operation, this setting is invalid and the register operates as a buffer register.

The TIOR registers are 8-bit registers that control the TGR registers. The TPU has eight TIOR registers, two each for channels 0 and 3, and one each for channels 1, 2, 4, and 5. The TIOR registers are initialized to H'00 by a reset, and in hardware standby mode.

Care is required since TIOR is affected by the TMDR setting. The initial output specified by TIOR is valid when the counter is stopped (the CST bit in TSTR is cleared to 0). Note also that, in PWM mode 2, the output at the point at which the counter is cleared to 0 is specified.

**Bits 7 to 4— I/O Control B3 to B0 (IOB3 to IOB0)**
**I/O Control D3 to D0 (IOD3 to IOD0):**
Bits IOB3 to IOB0 specify the function of TGRB.
Bits IOD3 to IOD0 specify the function of TGRD.

- TIOR0H

| Channel | Bit 7 IOB3 | Bit 6 IOB2 | Bit 5 IOB1 | Bit 4 IOB0 | Description | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | TGR0B is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR0B is input capture register | Capture input source is TIOCB0 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is channel 1/count clock | Input capture at TCNT1 count- up/count-down*[1] |

*: Don't care

Note: 1. When bits TPSC2 to TPSC0 in TCR1 are set to B'000 and ø/1 is used as the TCNT1 count clock, this setting is invalid and input capture is not generated.

- TIOR0L

| Channel | Bit 7 IOD3 | Bit 6 IOD2 | Bit 5 IOD1 | Bit 4 IOD0 | Description | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | TGR0D is output compare register*[2] | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR0D is input capture register*[2] | Capture input source is TIOCD0 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is channel 1/count clock | Input capture at TCNT1 count-up/count-down*[1] |

*: Don’t care

Notes: 1. When bits TPSC2 to TPSC0 in TCR1 are set to B'000 and ø/1 is used as the TCNT1 count clock, this setting is invalid and input capture is not generated.

2. When the BFB bit in TMDR0 is set to 1 and TGR0D is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

- TIOR1

| Channel | Bit 7 IOB3 | Bit 6 IOB2 | Bit 5 IOB1 | Bit 4 IOB0 | Description | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | TGR1B is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR1B is input capture register | Capture input source is TIOCB1 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is TGR0C compare match/ input capture | Input capture at generation of TGR0C compare match/input capture |

*: Don't care

- TIOR2

| Channel | Bit 7 IOB3 | Bit 6 IOB2 | Bit 5 IOB1 | Bit 4 IOB0 | Description | | |
|---|---|---|---|---|---|---|---|
| 2 | 0 | 0 | 0 | 0 | TGR2B is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | * | 0 | 0 | TGR2B is input capture register | Capture input source is TIOCB2 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |

*: Don’t care

- TIOR3H

| Channel | Bit 7 IOB3 | Bit 6 IOB2 | Bit 5 IOB1 | Bit 4 IOB0 | Description | | |
|---|---|---|---|---|---|---|---|
| 3 | 0 | 0 | 0 | 0 | TGR3B is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR3B is input capture register | Capture input source is TIOCB3 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is channel 4/count clock | Input capture at TCNT4 count-up/count-down*[1] |

*: Don't care

Note: 1. When bits TPSC2 to TPSC0 in TCR4 are set to B'000 and ø/1 is used as the TCNT4 count clock, this setting is invalid and input capture is not generated.

- TIOR3L

| Channel | Bit 7 IOD3 | Bit 6 IOD2 | Bit 5 IOD1 | Bit 4 IOD0 | Description | | |
|---|---|---|---|---|---|---|---|
| 3 | 0 | 0 | 0 | 0 | TGR3D is output compare register*[2] | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR3D is input capture register*[2] | Capture input source is TIOCD3 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is channel 4/count clock | Input capture at TCNT4 count-up/count-down*[1] |

*: Don’t care

Notes: 1. When bits TPSC2 to TPSC0 in TCR4 are set to B'000 and ø/1 is used as the TCNT4 count clock, this setting is invalid and input capture is not generated.

2. When the BFB bit in TMDR3 is set to 1 and TGR3D is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

- TIOR4

| Channel | Bit 7 IOB3 | Bit 6 IOB2 | Bit 5 IOB1 | Bit 4 IOB0 | Description | | |
|---|---|---|---|---|---|---|---|
| 4 | 0 | 0 | 0 | 0 | TGR4B is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR4B is input capture register | Capture input source is TIOCB4 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is TGR3C compare match/ input capture | Input capture at generation of TGR3C compare match/ input capture |

*: Don't care

- TIOR5

| Channel | Bit 7 IOB3 | Bit 6 IOB2 | Bit 5 IOB1 | Bit 4 IOB0 | Description | | |
|---|---|---|---|---|---|---|---|
| 5 | 0 | 0 | 0 | 0 | TGR5B is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | * | 0 | 0 | TGR5B is input capture register | Capture input source is TIOCB5 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |

*: Don’t care

**Bits 3 to 0— I/O Control A3 to A0 (IOA3 to IOA0)**
**I/O Control C3 to C0 (IOC3 to IOC0):**

IOA3 to IOA0 specify the function of TGRA.
IOC3 to IOC0 specify the function of TGRC.

- TIOR0H

| Channel | Bit 3 IOA3 | Bit 2 IOA2 | Bit 1 IOA1 | Bit 0 IOA0 | Description | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | TGR0A is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR0A is input capture register | Capture input source is TIOCA0 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is channel 1/ count clock | Input capture at TCNT1 count-up/count-down |

*: Don't care

- TIOR0L

| | Bit 3 | Bit 2 | Bit 1 | Bit 0 | | | |
|---|---|---|---|---|---|---|---|
| **Channel** | **IOC3** | **IOC2** | **IOC1** | **IOC0** | **Description** | | |
| 0 | 0 | 0 | 0 | 0 | TGR0C is output compare register*[1] | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR0C is input capture register*[1] | Capture input source is TIOCC0 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is channel 1/count clock | Input capture at TCNT1 count-up/count-down |

*: Don't care

Note: 1. When the BFA bit in TMDR0 is set to 1 and TGR0C is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

- TIOR1

| Channel | Bit 3 IOA3 | Bit 2 IOA2 | Bit 1 IOA1 | Bit 0 IOA0 | Description | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | TGR1A is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR1A is input capture register | Capture input source is TIOCA1 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is TGR0A compare match/ input capture | Input capture at generation of channel 0/TGR0A compare match/input capture |

*: Don’t care

- TIOR2

| Channel | Bit 3 IOA3 | Bit 2 IOA2 | Bit 1 IOA1 | Bit 0 IOA0 | Description | | |
|---|---|---|---|---|---|---|---|
| 2 | 0 | 0 | 0 | 0 | TGR2A is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | * | 0 | 0 | TGR2A is input capture register | Capture input source is TIOCA2 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |

*: Don’t care

- TIOR3H

| Channel | Bit 3 IOA3 | Bit 2 IOA2 | Bit 1 IOA1 | Bit 0 IOA0 | Description | | |
|---|---|---|---|---|---|---|---|
| 3 | 0 | 0 | 0 | 0 | TGR3A is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR3A is input capture register | Capture input source is TIOCA3 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is channel 4/count clock | Input capture at TCNT4 count-up/count-down |

*: Don’t care

- TIOR3L

| Channel | Bit 3 IOC3 | Bit 2 IOC2 | Bit 1 IOC1 | Bit 0 IOC0 | Description | | |
|---|---|---|---|---|---|---|---|
| 3 | 0 | 0 | 0 | 0 | TGR3C is output compare register*[1] | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR3C is input capture register*[1] | Capture input source is TIOCC3 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is channel 4/count clock | Input capture at TCNT4 count-up/count-down |

*: Don't care

Note: 1. When the BFA bit in TMDR3 is set to 1 and TGR3C is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

- TIOR4

| Channel | Bit 3 IOA3 | Bit 2 IOA2 | Bit 1 IOA1 | Bit 0 IOA0 | Description | | |
|---|---|---|---|---|---|---|---|
| 4 | 0 | 0 | 0 | 0 | TGR4A is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | 0 | TGR4A is input capture register | Capture input source is TIOCA4 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |
| | | 1 | * | * | | Capture input source is TGR3A compare match/ input capture | Input capture at generation of TGR3A compare match/input capture |

*: Don't care

- TIOR5

| Channel | Bit 3 IOA3 | Bit 2 IOA2 | Bit 1 IOA1 | Bit 0 IOA0 | Description | | |
|---|---|---|---|---|---|---|---|
| 5 | 0 | 0 | 0 | 0 | TGR5A is output compare register | Output disabled | (Initial value) |
| | | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | | 1 | 0 | 0 | | Output disabled | |
| | | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | | 1 | 0 | | | 1 output at compare match |
| | | | | 1 | | | Toggle output at compare match |
| | 1 | * | 0 | 0 | TGR5A is input capture register | Capture input source is TIOCA5 pin | Input capture at rising edge |
| | | | | 1 | | | Input capture at falling edge |
| | | | 1 | * | | | Input capture at both edges |

*: Don’t care

### 9.2.4 Timer Interrupt Enable Register (TIER)

**Channel 0: TIER0**

**Channel 3: TIER3**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | TTGE | — | — | TCIEV | TGIED | TGIEC | TGIEB | TGIEA |
| Initial value | : | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | — | — | R/W | R/W | R/W | R/W | R/W |

**Channel 1: TIER1**

**Channel 2: TIER2**

**Channel 4: TIER4**

**Channel 5: TIER5**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | TTGE | — | TCIEU | TCIEV | — | — | TGIEB | TGIEA |
| Initial value | : | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | — | R/W | R/W | — | — | R/W | R/W |

The TIER registers are 8-bit registers that control enabling or disabling of interrupt requests for each channel. The TPU has six TIER registers, one for each channel. The TIER registers are initialized to H'40 by a reset, and in hardware standby mode.

**Bit 7—A/D Conversion Start Request Enable (TTGE):** Enables or disables generation of A/D conversion start requests by TGRA input capture/compare match.

| Bit 7 | | |
|---|---|---|
| **TTGE** | **Description** | |
| 0 | A/D conversion start request generation disabled | (Initial value) |
| 1 | A/D conversion start request generation enabled | |

**Bit 6—Reserved:** Read-only bit, always read as 1.

**Bit 5—Underflow Interrupt Enable (TCIEU):** Enables or disables interrupt requests (TCIU) by the TCFU flag when the TCFU flag in TSR is set to 1 in channels 1 and 2.

In channels 0 and 3, bit 5 is reserved. It is always read as 0 and cannot be modified.

| Bit 5 | | |
|---|---|---|
| **TCIEU** | **Description** | |
| 0 | Interrupt requests (TCIU) by TCFU disabled | (Initial value) |
| 1 | Interrupt requests (TCIU) by TCFU enabled | |

**Bit 4—Overflow Interrupt Enable (TCIEV):** Enables or disables interrupt requests (TCIV) by the TCFV flag when the TCFV flag in TSR is set to 1.

| Bit 4 | | |
|---|---|---|
| **TCIEV** | **Description** | |
| 0 | Interrupt requests (TCIV) by TCFV disabled | (Initial value) |
| 1 | Interrupt requests (TCIV) by TCFV enabled | |

**Bit 3—TGR Interrupt Enable D (TGIED):** Enables or disables interrupt requests (TGID) by the TGFD bit when the TGFD bit in TSR is set to 1 in channels 0 and 3.

In channels 1, 2, 4, and 5, bit 3 is reserved. It is always read as 0 and cannot be modified.

| Bit 3 | | |
|---|---|---|
| **TGIED** | **Description** | |
| 0 | Interrupt requests (TGID) by TGFD bit disabled | (Initial value) |
| 1 | Interrupt requests (TGID) by TGFD bit enabled | |

**Bit 2—TGR Interrupt Enable C (TGIEC):** Enables or disables interrupt requests (TGIC) by the TGFC bit when the TGFC bit in TSR is set to 1 in channels 0 and 3.

In channels 1, 2, 4, and 5, bit 2 is reserved. It is always read as 0 and cannot be modified.

**Bit 2**

| TGIEC | Description | |
|---|---|---|
| 0 | Interrupt requests (TGIC) by TGFC bit disabled | (Initial value) |
| 1 | Interrupt requests (TGIC) by TGFC bit enabled | |

**Bit 1—TGR Interrupt Enable B (TGIEB):** Enables or disables interrupt requests (TGIB) by the TGFB bit when the TGFB bit in TSR is set to 1.

**Bit 1**

| TGIEB | Description | |
|---|---|---|
| 0 | Interrupt requests (TGIB) by TGFB bit disabled | (Initial value) |
| 1 | Interrupt requests (TGIB) by TGFB bit enabled | |

**Bit 0—TGR Interrupt Enable A (TGIEA):** Enables or disables interrupt requests (TGIA) by the TGFA bit when the TGFA bit in TSR is set to 1.

**Bit 0**

| TGIEA | Description | |
|---|---|---|
| 0 | Interrupt requests (TGIA) by TGFA bit disabled | (Initial value) |
| 1 | Interrupt requests (TGIA) by TGFA bit enabled | |

### 9.2.5 Timer Status Register (TSR)

**Channel 0: TSR0**

**Channel 3: TSR3**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | TCFV | TGFD | TGFC | TGFB | TGFA |
| Initial value | : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | — | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* |

Note: * Can only be written with 0 for flag clearing.

**Channel 1: TSR1**

**Channel 2: TSR2**

**Channel 4: TSR4**

**Channel 5: TSR5**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA |
| Initial value | : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R | — | R/(W)* | R/(W)* | — | — | R/(W)* | R/(W)* |

Note: * Can only be written with 0 for flag clearing.

The TSR registers are 8-bit registers that indicate the status of each channel. The TPU has six TSR registers, one for each channel. The TSR registers are initialized to H'C0 by a reset, and in hardware standby mode.

**Bit 7—Count Direction Flag (TCFD):** Status flag that shows the direction in which TCNT counts in channels 1, 2, 4, and 5.

In channels 0 and 3, bit 7 is reserved. It is always read as 1 and cannot be modified.

**Bit 7**

| TCFD | Description | |
|---|---|---|
| 0 | TCNT counts down | |
| 1 | TCNT counts up | (Initial value) |

**Bit 6—Reserved:** Read-only bit, always read as 1.

**Bit 5—Underflow Flag (TCFU):** Status flag that indicates that TCNT underflow has occurred when channels 1, 2, 4, and 5 are set to phase counting mode.

In channels 0 and 3, bit 5 is reserved. It is always read as 0 and cannot be modified.

**Bit 5**

| TCFU | Description | |
|---|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFU after reading TCFU = 1 | (Initial value) |
| 1 | [Setting condition]<br>When the TCNT value underflows (changes from H'0000 to H'FFFF) | |

**Bit 4—Overflow Flag (TCFV):** Status flag that indicates that TCNT overflow has occurred.

**Bit 4**

| TCFV | Description | |
|---|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFV after reading TCFV = 1 | (Initial value) |
| 1 | [Setting condition]<br>When the TCNT value overflows (changes from H'FFFF to H'0000 ) | |

**Bit 3—Input Capture/Output Compare Flag D (TGFD):** Status flag that indicates the occurrence of TGRD input capture or compare match in channels 0 and 3.

In channels 1, 2, 4, and 5, bit 3 is reserved. It is always read as 0 and cannot be modified.

**Bit 3**

| TGFD | Description | |
|---|---|---|
| 0 | [Clearing conditions]<br>• When DTC is activated by TGID interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFD after reading TGFD = 1 | (Initial value) |
| 1 | [Setting conditions]<br>• When TCNT = TGRD while TGRD is functioning as output compare register<br>• When TCNT value is transferred to TGRD by input capture signal while TGRD is functioning as input capture register | |

**Bit 2—Input Capture/Output Compare Flag C (TGFC):** Status flag that indicates the occurrence of TGRC input capture or compare match in channels 0 and 3.

In channels 1, 2, 4, and 5, bit 2 is reserved. It is always read as 0 and cannot be modified.

**Bit 2**

| TGFC | Description |
|---|---|
| 0 | [Clearing conditions] (Initial value)<br>• When DTC is activated by TGIC interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFC after reading TGFC = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRC while TGRC is functioning as output compare register<br>• When TCNT value is transferred to TGRC by input capture signal while TGRC is functioning as input capture register |

**Bit 1—Input Capture/Output Compare Flag B (TGFB):** Status flag that indicates the occurrence of TGRB input capture or compare match.

**Bit 1**

| TGFB | Description |
|---|---|
| 0 | [Clearing conditions] (Initial value)<br>• When DTC is activated by TGIB interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFB after reading TGFB = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRB while TGRB is functioning as output compare register<br>• When TCNT value is transferred to TGRB by input capture signal while TGRB is functioning as input capture register |

**Bit 0—Input Capture/Output Compare Flag A (TGFA):** Status flag that indicates the occurrence of TGRA input capture or compare match.

**Bit 0**

| TGFA | Description |
|---|---|
| 0 | [Clearing conditions] (Initial value)<br>• When DTC is activated by TGIA interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFA after reading TGFA = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRA while TGRA is functioning as output compare register<br>• When TCNT value is transferred to TGRA by input capture signal while TGRA is functioning as input capture register |

### 9.2.6 Timer Counter (TCNT)

**Channel 0: TCNT0 (up-counter)**

**Channel 1: TCNT1 (up/down-counter*)**

**Channel 2: TCNT2 (up/down-counter*)**

**Channel 3: TCNT3 (up-counter)**

**Channel 4: TCNT4 (up/down-counter*)**

**Channel 5: TCNT5 (up/down-counter*)**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note : * These counters can be used as up/down-counters only in phase counting mode or when counting overflow/underflow on another channel. In other cases they function as up-counters.

The TCNT registers are 16-bit counters. The TPU has six TCNT counters, one for each channel.

The TCNT counters are initialized to H'0000 by a reset, and in hardware standby mode.

The TCNT counters cannot be accessed in 8-bit units; they must always be accessed as a 16-bit unit.

### 9.2.7 Timer General Register (TGR)

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The TGR registers are 16-bit registers with a dual function as output compare and input capture registers. The TPU has 16 TGR registers, four each for channels 0 and 3 and two each for channels 1, 2, 4, and 5. TGRC and TGRD for channels 0 and 3 can also be designated for operation as buffer registers*. The TGR registers are initialized to H'FFFF by a reset, and in hardware standby mode.

The TGR registers cannot be accessed in 8-bit units; they must always be accessed as a 16-bit unit.

Note: * TGR buffer register combinations are TGRA—TGRC and TGRB—TGRD.

### 9.2.8 Timer Start Register (TSTR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | CST5 | CST4 | CST3 | CST2 | CST1 | CST0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

TSTR is an 8-bit readable/writable register that selects operation/stoppage for channels 0 to 5. TSTR is initialized to H'00 by a reset, and in hardware standby mode.

Note: When setting the operating mode in TMDR or setting the count clock in TCR, first stop the TCNT counter.

**Bits 7 and 6—Reserved:** Should always be written with 0.

**Bits 5 to 0—Counter Start 5 to 0 (CST5 to CST0):** These bits select operation or stoppage for TCNT.

**Bit n**

| CSTn | Description | |
|---|---|---|
| 0 | TCNTn count operation is stopped | (Initial value) |
| 1 | TCNTn performs count operation | |

n = 5 to 0

Note: If 0 is written to the CST bit during operation with the TIOC pin designated for output, the counter stops but the TIOC pin output compare output level is retained. If TIOR is written to when the CST bit is cleared to 0, the pin output level will be changed to the set initial output value.

### 9.2.9 Timer Synchro Register (TSYR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | SYNC5 | SYNC4 | SYNC3 | SYNC2 | SYNC1 | SYNC0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

TSYR is an 8-bit readable/writable register that selects independent operation or synchronous operation for the channel 0 to 5 TCNT counters. A channel performs synchronous operation when the corresponding bit in TSYR is set to 1.

TSYR is initialized to H'00 by a reset, and in hardware standby mode.

**Bits 7 and 6—Reserved:** Should always be written with 0.

**Bits 5 to 0—Timer Synchro 5 to 0 (SYNC5 to SYNC0):** These bits select whether operation is independent of or synchronized with other channels.

When synchronous operation is selected, synchronous presetting of multiple channels*[1], and synchronous clearing through counter clearing on another channel*[2] are possible.

**Bit n**

| SYNCn | Description |
|---|---|
| 0 | TCNTn operates independently (TCNT presetting/clearing is unrelated to other channels) (Initial value) |
| 1 | TCNTn performs synchronous operation<br>TCNT synchronous presetting/synchronous clearing is possible |

n = 5 to 0

Notes: 1. To set synchronous operation, the SYNC bits for at least two channels must be set to 1.
2. To set synchronous clearing, in addition to the SYNC bit , the TCNT clearing source must also be set by means of bits CCLR2 to CCLR0 in TCR.

### 9.2.10 Module Stop Control Register (MSTPCR)

| | | MSTPCRH | | | | | | | | MSTPCRL | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | | |
| Initial value | : | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the MSTP13 bit in MSTPCR is set to 1, TPU operation stops at the end of the bus cycle and a transition is made to module stop mode. Registers cannot be read or written to in module stop mode. For details, see section 19.5, Module Stop Mode.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 13—Module Stop (MSTP13):** Specifies the TPU module stop mode.

**Bit 13**

| MSTP13 | Description | |
|---|---|---|
| 0 | TPU module stop mode cleared | |
| 1 | TPU module stop mode set | (Initial value) |

## 9.3 Interface to Bus Master

### 9.3.1 16-Bit Registers

TCNT and TGR are 16-bit registers. As the data bus to the bus master is 16 bits wide, these registers can be read and written to in 16-bit units.

These registers cannot be read or written to in 8-bit units; 16-bit access must always be used.

An example of 16-bit register access operation is shown in figure 9.2.

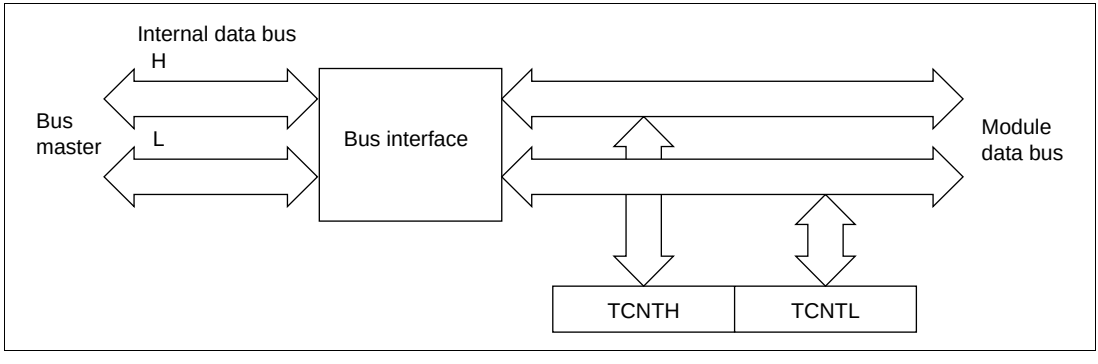



**Figure 9.2 16-Bit Register Access Operation [Bus Master ↔ TCNT (16 Bits)]**

### 9.3.2 8-Bit Registers

Registers other than TCNT and TGR are 8-bit. As the data bus to the CPU is 16 bits wide, these registers can be read and written to in 16-bit units. They can also be read and written to in 8-bit units.

Examples of 8-bit register access operation are shown in figures 9.3, 9.4, and 9.5.

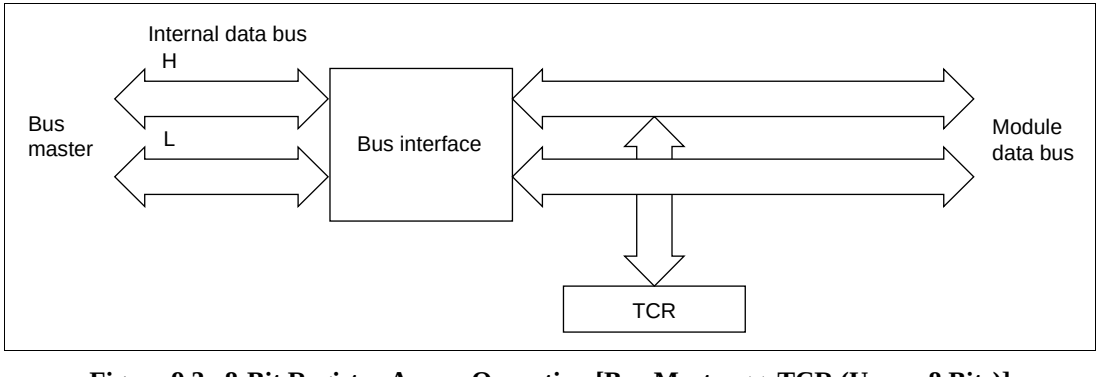



**Figure 9.3 8-Bit Register Access Operation [Bus Master ↔ TCR (Upper 8 Bits)]**

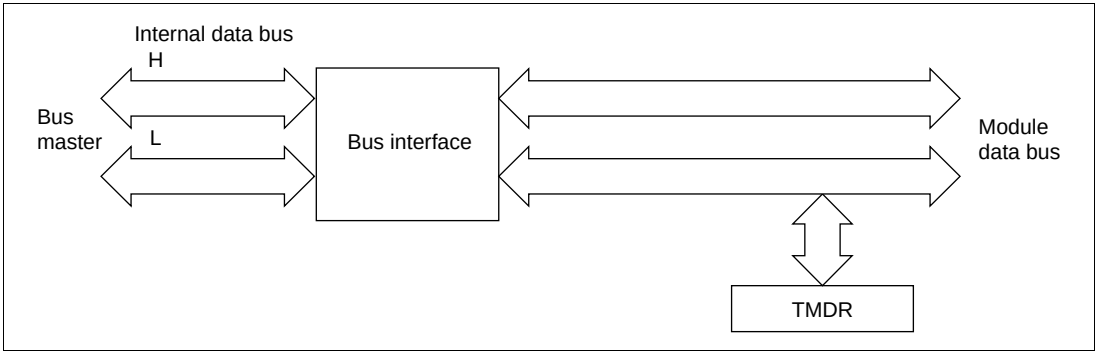



**Figure 9.4 8-Bit Register Access Operation [Bus Master ↔ TMDR (Lower 8 Bits)]**

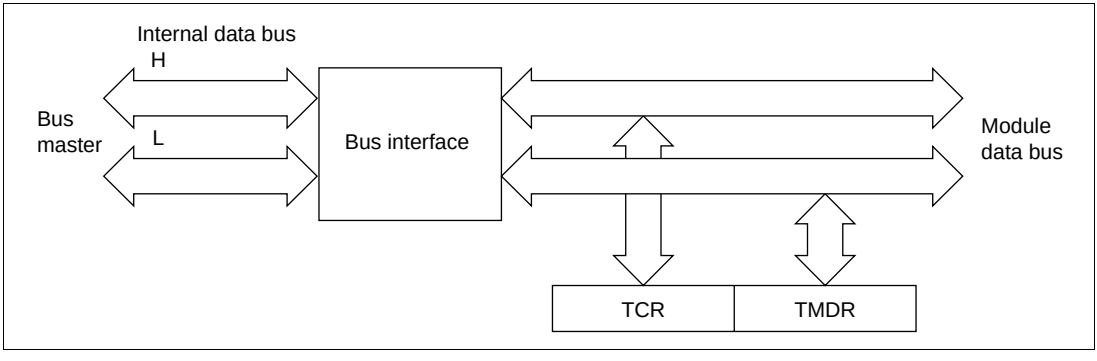



**Figure 9.5 8-Bit Register Access Operation [Bus Master ↔ TCR and TMDR (16 Bits)]**

## 9.4 Operation

### 9.4.1 Overview

Operation in each mode is outlined below.

**Normal Operation:** Each channel has a TCNT and TGR register. TCNT performs up-counting, and is also capable of free-running operation, synchronous counting, and external event counting.

Each TGR can be used as an input capture register or output compare register.

**Synchronous Operation:** When synchronous operation is designated for a channel, TCNT for that channel performs synchronous presetting. That is, when TCNT for a channel designated for synchronous operation is rewritten, the TCNT counters for the other channels are also rewritten at the same time. Synchronous clearing of the TCNT counters is also possible by setting the timer synchronization bits in TSYR for channels designated for synchronous operation.

**Buffer Operation**

- When TGR is an output compare register

  When a compare match occurs, the value in the buffer register for the relevant channel is transferred to TGR.

- When TGR is an input capture register

  When input capture occurs, the value in TCNT is transfer to TGR and the value previously held in TGR is transferred to the buffer register.

**Cascaded Operation:** The channel 1 counter (TCNT1), channel 2 counter (TCNT2), channel 4 counter (TCNT4), and channel 5 counter (TCNT5) can be connected together to operate as a 32-bit counter.

**PWM Mode:** In this mode, a PWM waveform is output. The output level can be set by means of TIOR. A PWM waveform with a duty of between 0% and 100% can be output, according to the setting of each TGR register.

**Phase Counting Mode:** In this mode, TCNT is incremented or decremented by detecting the phases of two clocks input from the external clock input pins in channels 1, 2, 4, and 5. When phase counting mode is set, the corresponding TCLK pin functions as the clock pin, and TCNT performs up- or down-counting.

This can be used for two-phase encoder pulse input.

### 9.4.2 Basic Functions

**Counter Operation:** When one of bits CST0 to CST5 is set to 1 in TSTR, the TCNT counter for the corresponding channel starts counting. TCNT can operate as a free-running counter, periodic counter, and so on.

- Example of count operation setting procedure
  Figure 9.6 shows an example of the count operation setting procedure.

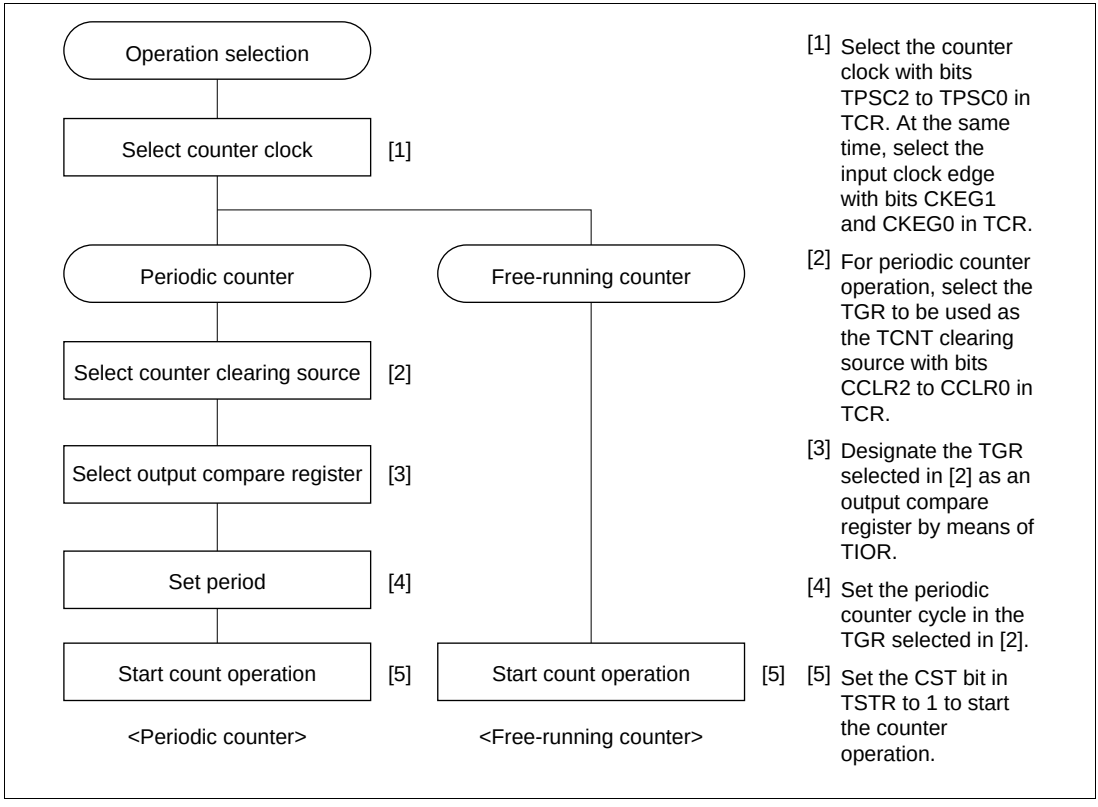



**Figure 9.6 Example of Counter Operation Setting Procedure**

- Free-running count operation and periodic count operation

  Immediately after a reset, the TPU's TCNT counters are all designated as free-running counters. When the relevant bit in TSTR is set to 1 the corresponding TCNT counter starts up-count operation as a free-running counter. When TCNT overflows (from H'FFFF to H'0000), the TCFV bit in TSR is set to 1. If the value of the corresponding TCIEV bit in TIER is 1 at this point, the TPU requests an interrupt. After overflow, TCNT starts counting up again from H'0000.

  Figure 9.7 illustrates free-running counter operation.

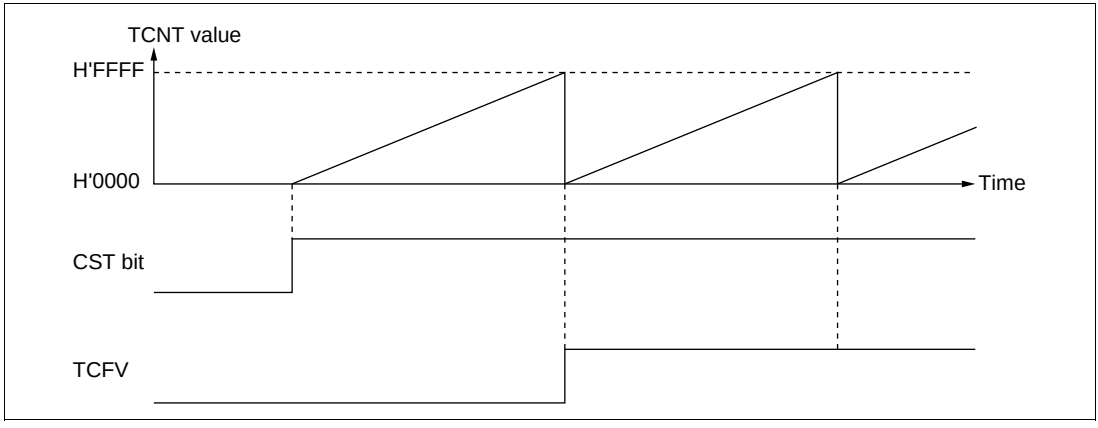



**Figure 9.7 Free-Running Counter Operation**

When compare match is selected as the TCNT clearing source, the TCNT counter for the relevant channel performs periodic count operation. The TGR register for setting the period is designated as an output compare register, and counter clearing by compare match is selected by means of bits CCLR2 to CCLR0 in TCR. After the settings have been made, TCNT starts up-count operation as periodic counter when the corresponding bit in TSTR is set to 1. When the count value matches the value in TGR, the TGF bit in TSR is set to 1 and TCNT is cleared to H'0000.

If the value of the corresponding TGIE bit in TIER is 1 at this point, the TPU requests an interrupt. After a compare match, TCNT starts counting up again from H'0000.

Figure 9.8 illustrates periodic counter operation.

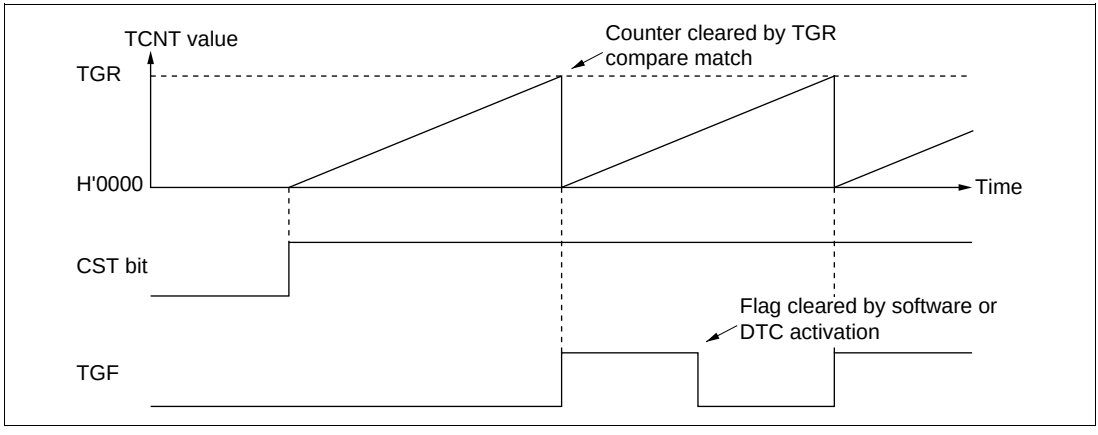



**Figure 9.8 Periodic Counter Operation**

**Waveform Output by Compare Match:** The TPU can perform 0, 1, or toggle output from the corresponding output pin using compare match.

- Example of setting procedure for waveform output by compare match
  Figure 9.9 shows an example of the setting procedure for waveform output by compare match

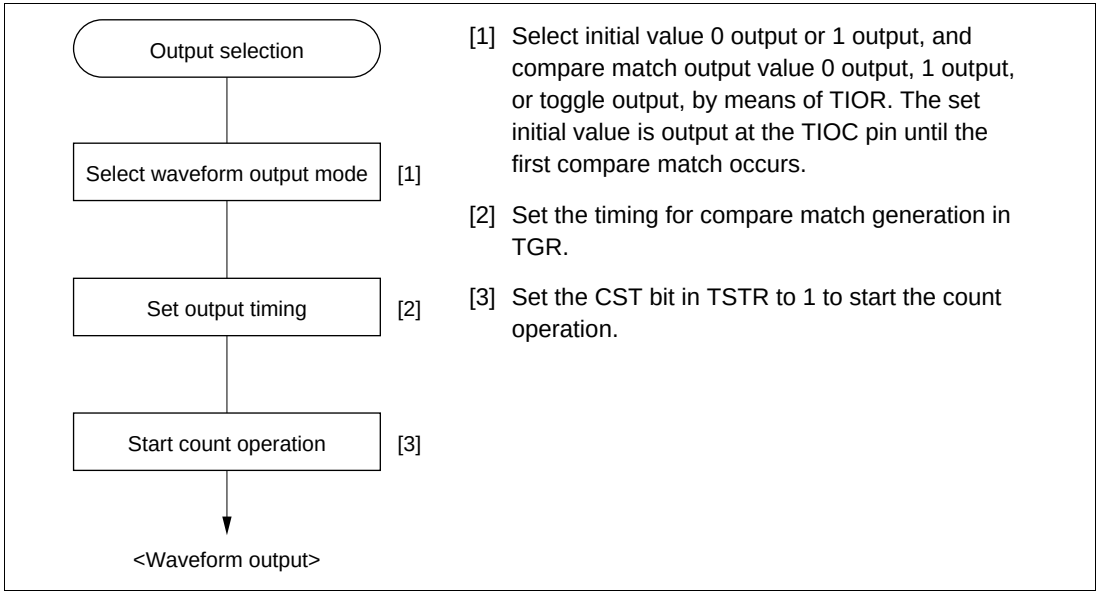



**Figure 9.9 Example Of Setting Procedure For Waveform Output By Compare Match**

- Examples of waveform output operation

  Figure 9.10 shows an example of 0 output/1 output.

  In this example TCNT has been designated as a free-running counter, and settings have been made so that 1 is output by compare match A, and 0 is output by compare match B. When the set level and the pin level coincide, the pin level does not change.

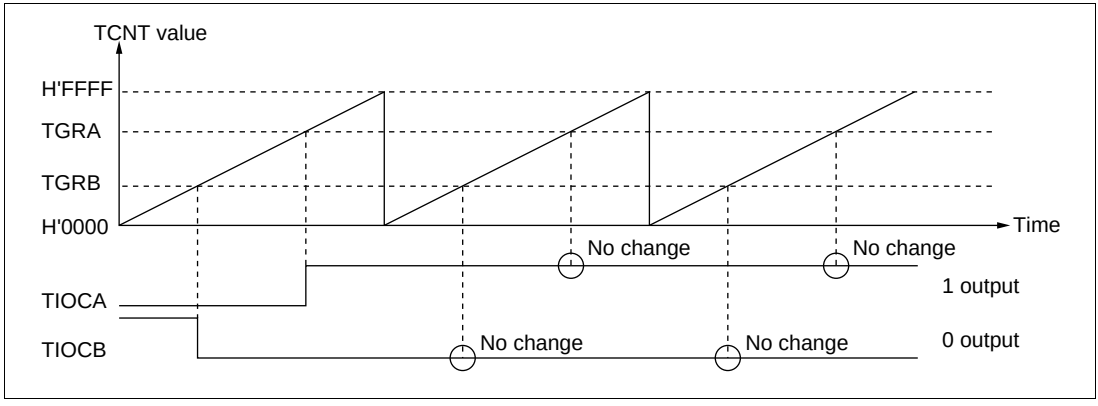



**Figure 9.10 Example of 0 Output/1 Output Operation**

Figure 9.11 shows an example of toggle output.

In this example TCNT has been designated as a periodic counter (with counter clearing performed by compare match B), and settings have been made so that output is toggled by both compare match A and compare match B.

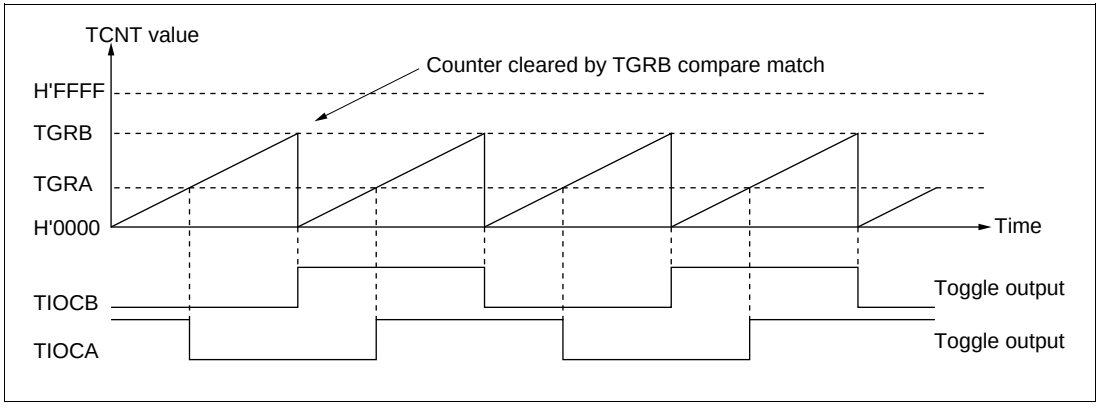


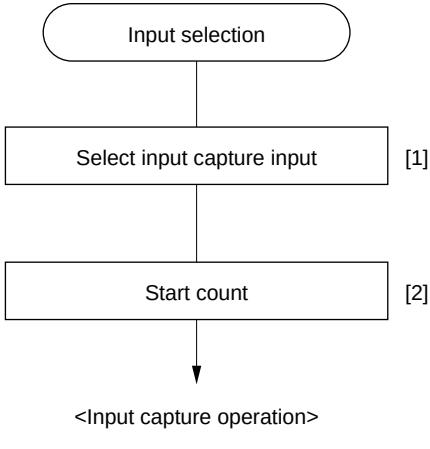

**Figure 9.11 Example of Toggle Output Operation**

**Input Capture Function:** The TCNT value can be transferred to TGR on detection of the TIOC pin input edge.

Rising edge, falling edge, or both edges can be selected as the detected edge. For channels 0, 1, 3, and 4, it is also possible to specify another channel's counter input clock or compare match signal as the input capture source.

Note: When another channel's counter input clock is used as the input capture input for channels 0 and 3, ø/1 should not be selected as the counter input clock used for input capture input. Input capture will not be generated if ø/1 is selected.

- Example of input capture operation setting procedure
  Figure 9.12 shows an example of the input capture operation setting procedure.




[1] Designate TGR as an input capture register by means of TIOR, and select rising edge, falling edge, or both edges as the input capture source and input signal edge.

[2] Set the CST bit in TSTR to 1 to start the count operation.

**Figure 9.12 Example of Input Capture Operation Setting Procedure**

- Example of input capture operation

  Figure 9.13 shows an example of input capture operation.

  In this example both rising and falling edges have been selected as the TIOCA pin input capture input edge, falling edge has been selected as the TIOCB pin input capture input edge, and counter clearing by TGRB input capture has been designated for TCNT.

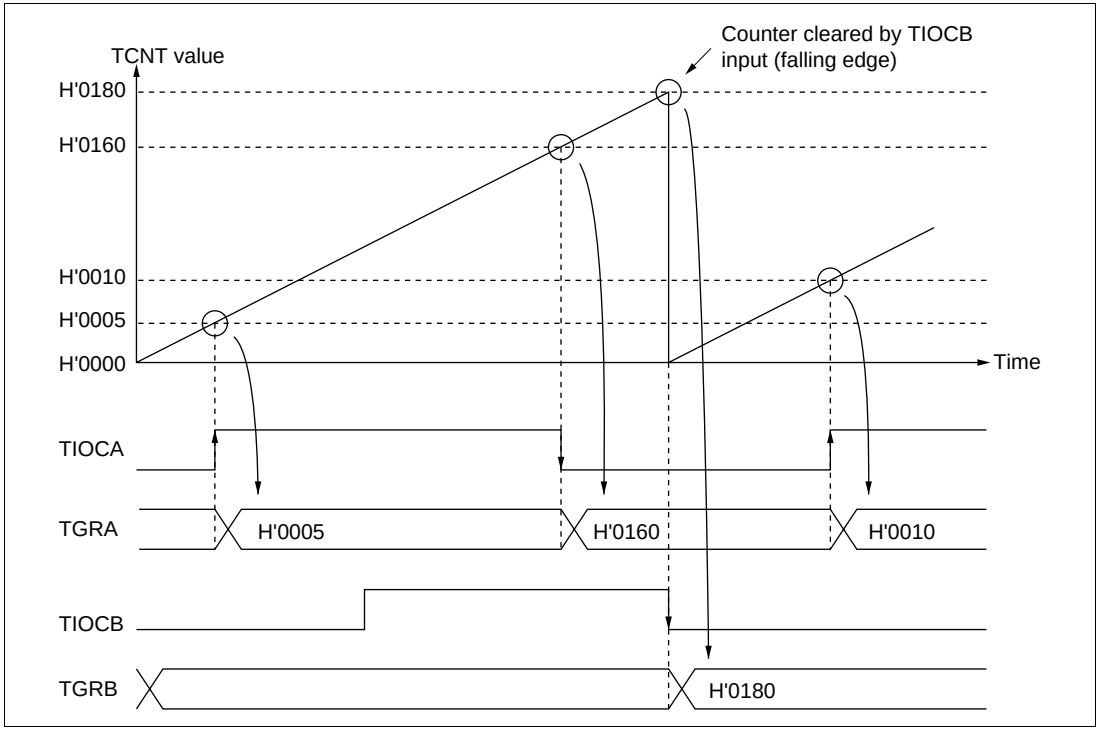



**Figure 9.13 Example of Input Capture Operation**

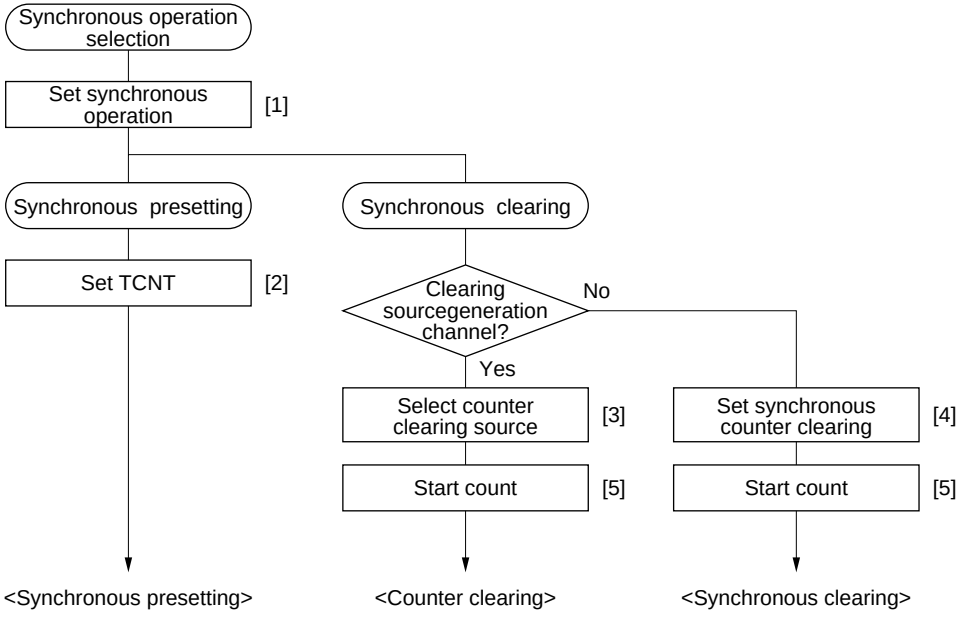
### 9.4.3 Synchronous Operation

In synchronous operation, the values in a number of TCNT counters can be rewritten simultaneously (synchronous presetting). Also, a number of TCNT counters can be cleared simultaneously by making the appropriate setting in TCR (synchronous clearing).

Synchronous operation enables TGR to be incremented with respect to a single time base.

Channels 0 to 5 can all be designated for synchronous operation.

**Example of Synchronous Operation Setting Procedure:** Figure 9.14 shows an example of the synchronous operation setting procedure.




[1] Set to 1 the SYNC bits in TSYR corresponding to the channels to be designated for synchronous operation.

[2] When the TCNT counter of any of the channels designated for synchronous operation is written to, the same value is simultaneously written to the other TCNT counters.

[3] Use bits CCLR2 to CCLR0 in TCR to specify TCNT clearing by input capture/output compare, etc.

[4] Use bits CCLR2 to CCLR0 in TCR to designate synchronous clearing for the counter clearing source.

[5] Set to 1 the CST bits in TSTR for the relevant channels, to start the count operation.

**Figure 9.14 Example of Synchronous Operation Setting Procedure**

**Example of Synchronous Operation:** Figure 9.15 shows an example of synchronous operation.

In this example, synchronous operation and PWM mode 1 have been designated for channels 0 to 2, TGR0B compare match has been set as the channel 0 counter clearing source, and synchronous clearing has been set for the channel 1 and 2 counter clearing source.

Three-phase PWM waveforms are output from pins TIOC0A, TIOC1A, and TIOC2A. At this time, synchronous presetting, and synchronous clearing by TGR0B compare match, is performed for channel 0 to 2 TCNT counters, and the data set in TGR0B is used as the PWM cycle.

For details of PWM modes, see section 9.4.6, PWM Modes.

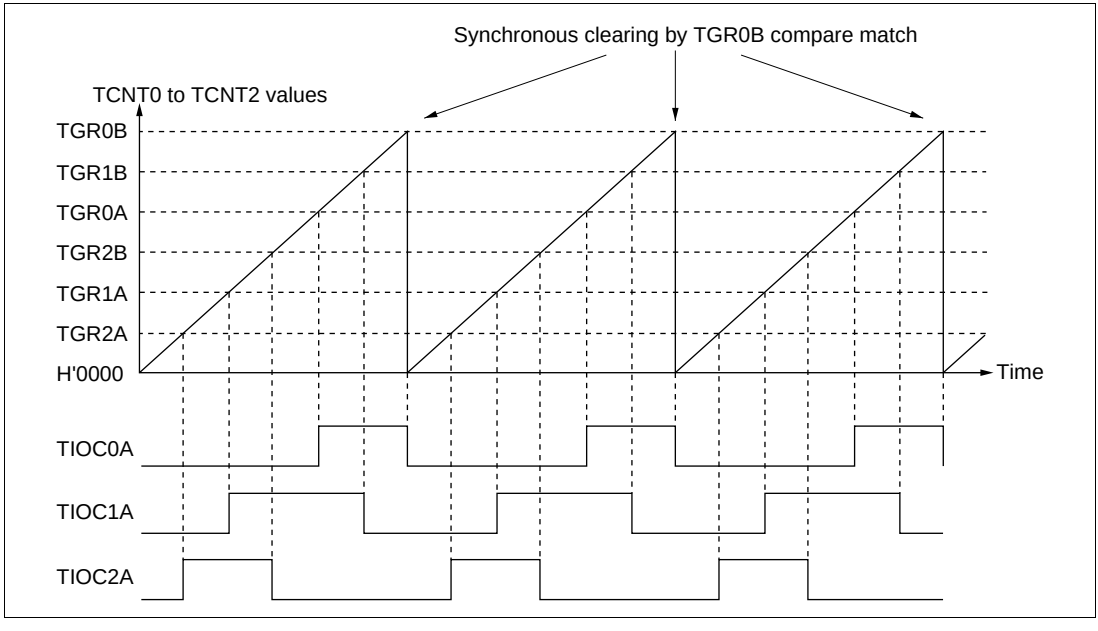



**Figure 9.15 Example of Synchronous Operation**

### 9.4.4 Buffer Operation

Buffer operation, provided for channels 0 and 3, enables TGRC and TGRD to be used as buffer registers.

Buffer operation differs depending on whether TGR has been designated as an input capture register or as a compare match register.

Table 9.5 shows the register combinations used in buffer operation.

**Table 9.5 Register Combinations in Buffer Operation**

| Channel | Timer General Register | Buffer Register |
|---|---|---|
| 0 | TGR0A | TGR0C |
| | TGR0B | TGR0D |
| 3 | TGR3A | TGR3C |
| | TGR3B | TGR3D |

- When TGR is an output compare register

  When a compare match occurs, the value in the buffer register for the corresponding channel is transferred to the timer general register.

  This operation is illustrated in figure 9.16.

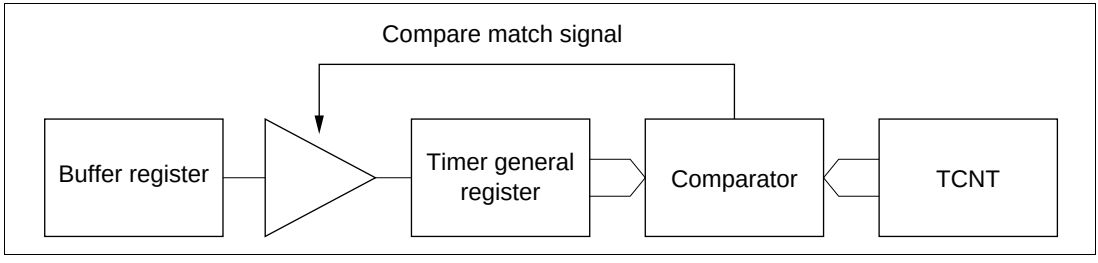



**Figure 9.16 Compare Match Buffer Operation**

- When TGR is an input capture register

  When input capture occurs, the value in TCNT is transferred to TGR and the value previously held in the timer general register is transferred to the buffer register.

  This operation is illustrated in figure 9.17.

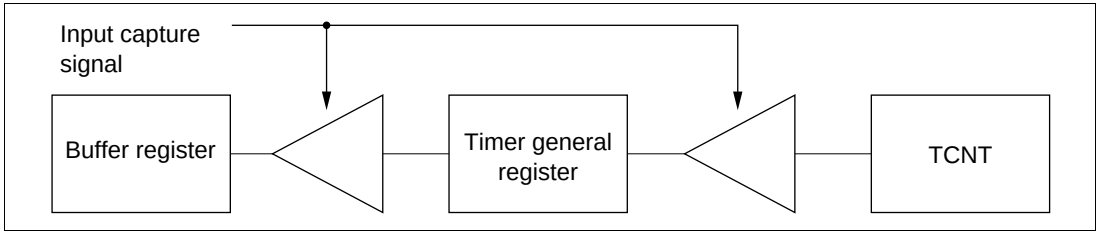



**Figure 9.17 Input Capture Buffer Operation**

**Example of Buffer Operation Setting Procedure:** Figure 9.18 shows an example of the buffer operation setting procedure.

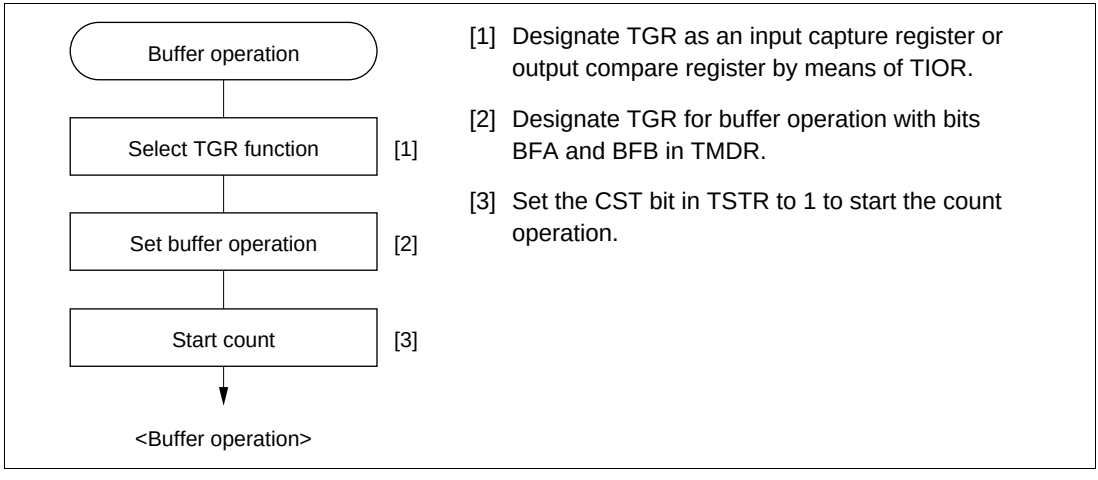



**Figure 9.18 Example of Buffer Operation Setting Procedure**

**Examples of Buffer Operation**

- When TGR is an output compare register

  Figure 9.19 shows an operation example in which PWM mode 1 has been designated for channel 0, and buffer operation has been designated for TGRA and TGRC. The settings used in this example are TCNT clearing by compare match B, 1 output at compare match A, and 0 output at compare match B.

  As buffer operation has been set, when compare match A occurs the output changes and the value in buffer register TGRC is simultaneously transferred to timer general register TGRA. This operation is repeated each time compare match A occurs.

  For details of PWM modes, see section 9.4.6, PWM Modes.

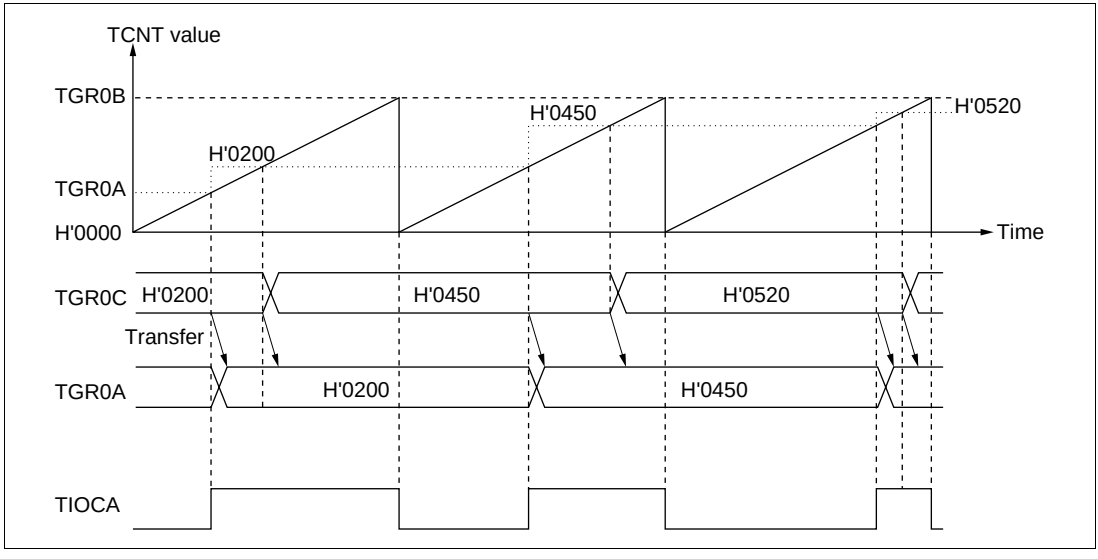



**Figure 9.19 Example of Buffer Operation (1)**

- When TGR is an input capture register

  Figure 9.20 shows an operation example in which TGRA has been designated as an input capture register, and buffer operation has been designated for TGRA and TGRC.

  Counter clearing by TGRA input capture has been set for TCNT, and both rising and falling edges have been selected as the TIOCA pin input capture input edge.

  As buffer operation has been set, when the TCNT value is stored in TGRA upon occurrence of input capture A, the value previously stored in TGRA is simultaneously transferred to TGRC.

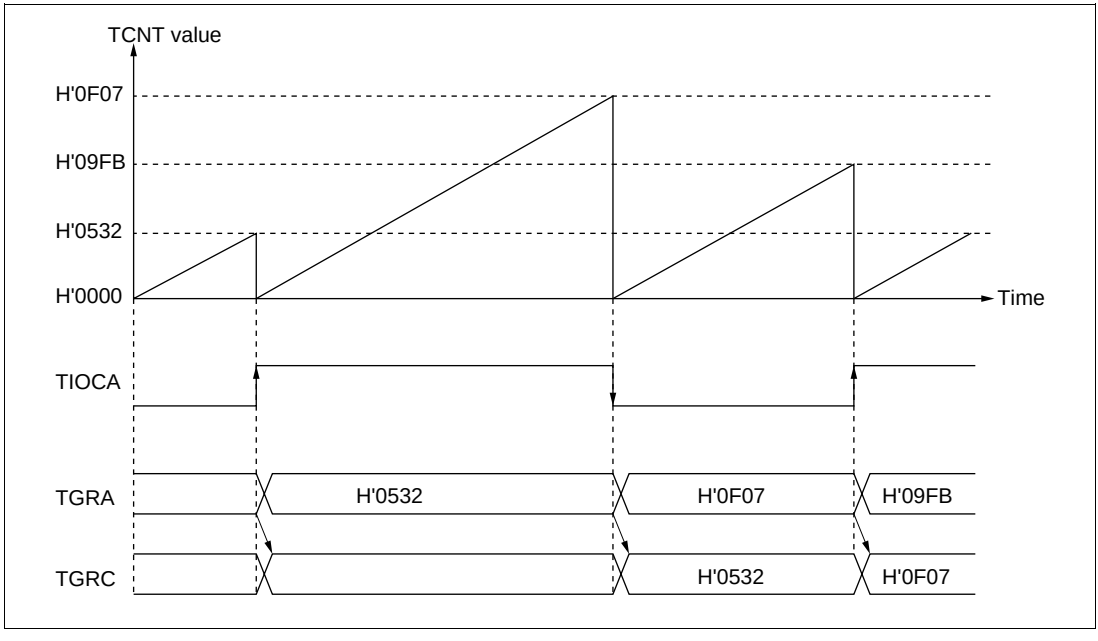



**Figure 9.20 Example of Buffer Operation (2)**

### 9.4.5 Cascaded Operation

In cascaded operation, two 16-bit counters for different channels are used together as a 32-bit counter.

This function works by counting the channel 1 (channel 4) counter clock upon overflow/underflow of TCNT2 (TCNT5) as set in bits TPSC2 to TPSC0 in TCR.

Underflow occurs only when the lower 16-bit TCNT is in phase-counting mode.

Table 9.6 shows the register combinations used in cascaded operation.

Note: When phase counting mode is set for channel 1 or 4, the counter clock setting is invalid and the counter operates independently in phase counting mode.

**Table 9.6 Cascaded Combinations**

| Combination | Upper 16 Bits | Lower 16 Bits |
|---|---|---|
| Channels 1 and 2 | TCNT1 | TCNT2 |
| Channels 4 and 5 | TCNT4 | TCNT5 |

**Example of Cascaded Operation Setting Procedure:** Figure 9.21 shows an example of the setting procedure for cascaded operation.

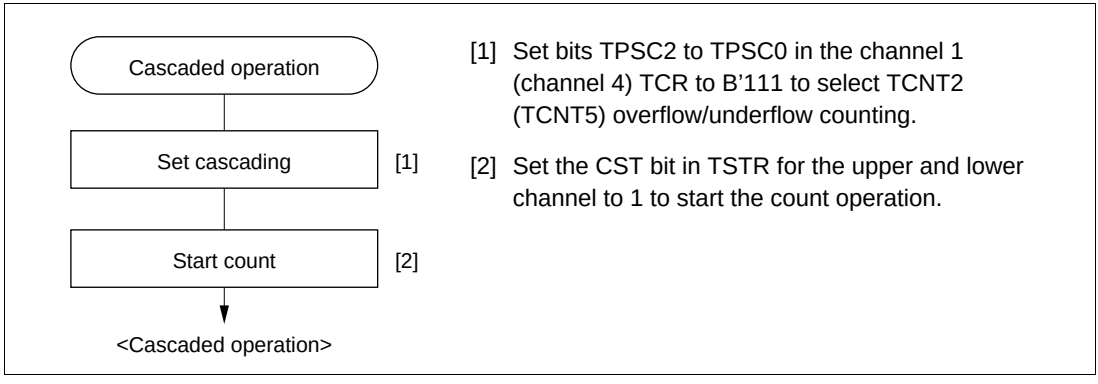



**Figure 9.21 Cascaded Operation Setting Procedure**

**Examples of Cascaded Operation:** Figure 9.22 illustrates the operation when counting upon TCNT2 overflow/underflow has been set for TCNT1, TGR1A and TGR2A have been designated as input capture registers, and TIOC pin rising edge has been selected.

When a rising edge is input to the TIOCA1 and TIOCA2 pins simultaneously, the upper 16 bits of the 32-bit data are transferred to TGR1A, and the lower 16 bits to TGR2A.

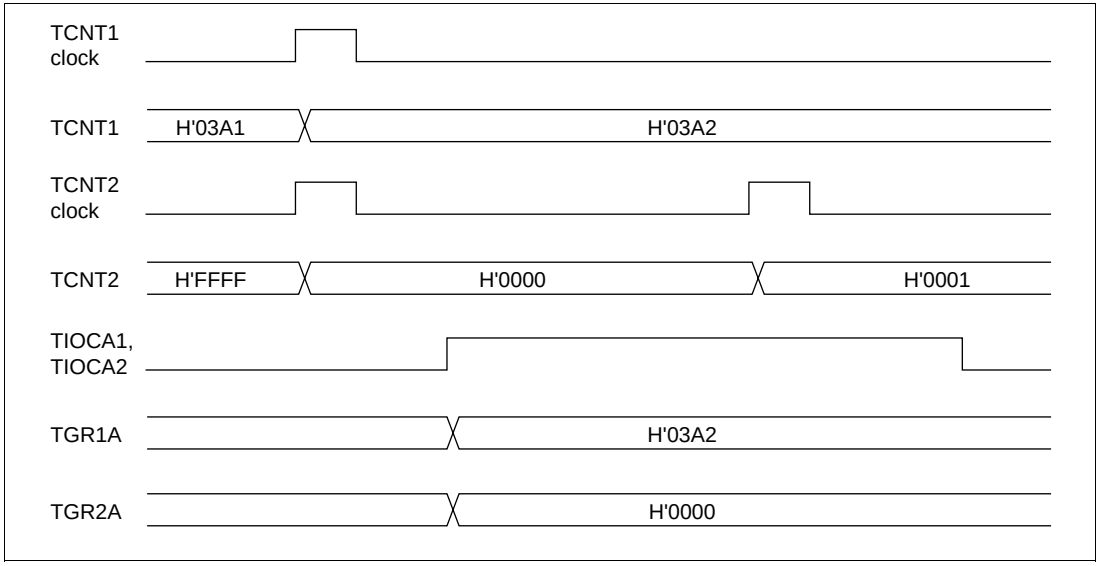



**Figure 9.22 Example of Cascaded Operation (1)**

Figure 9.23 illustrates the operation when counting upon TCNT2 overflow/underflow has been set for TCNT1, and phase counting mode has been designated for channel 2.

TCNT1 is incremented by TCNT2 overflow and decremented by TCNT2 underflow.

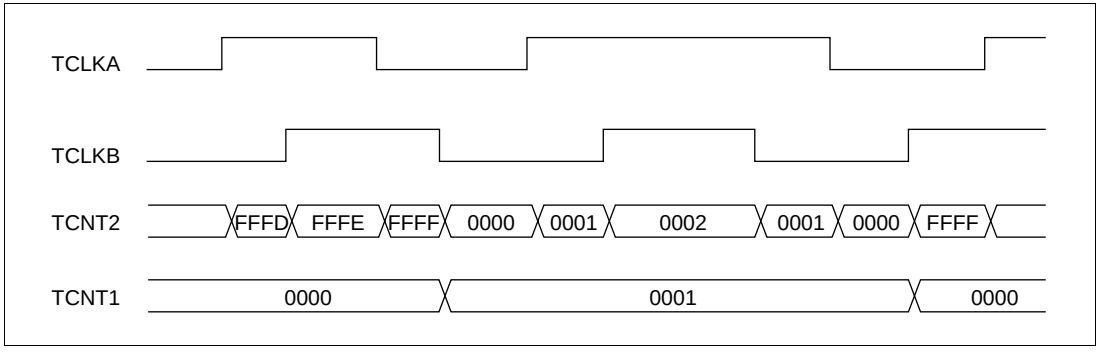



**Figure 9.23 Example of Cascaded Operation (2)**

### 9.4.6 PWM Modes

In PWM mode, PWM waveforms are output from the output pins. 0, 1, or toggle output can be selected as the output level in response to compare match of each TGR.

Designating TGR compare match as the counter clearing source enables the period to be set in that register. All channels can be designated for PWM mode independently. Synchronous operation is also possible.

There are two PWM modes, as described below.

- PWM mode 1

  PWM output is generated from the TIOCA and TIOCC pins by pairing TGRA with TGRB and TGRC with TGRD. The output specified by bits IOA3 to IOA0 and IOC3 to IOC0 in TIOR is output from the TIOCA and TIOCC pins at compare matches A and C, and the output specified by bits IOB3 to IOB0 and IOD3 to IOD0 in TIOR is output at compare matches B and D. The initial output value is the value set in TGRA or TGRC. If the set values of paired TGRs are identical, the output value does not change when a compare match occurs.

  In PWM mode 1, a maximum 8-phase PWM output is possible.

- PWM mode 2

  PWM output is generated using one TGR as the cycle register and the others as duty registers. The output specified in TIOR is performed by means of compare matches. Upon counter clearing by a synchronization register compare match, the output value of each pin is the initial value set in TIOR. If the set values of the cycle and duty registers are identical, the output value does not change when a compare match occurs.

  In PWM mode 2, a maximum 15-phase PWM output is possible by combined use with synchronous operation.

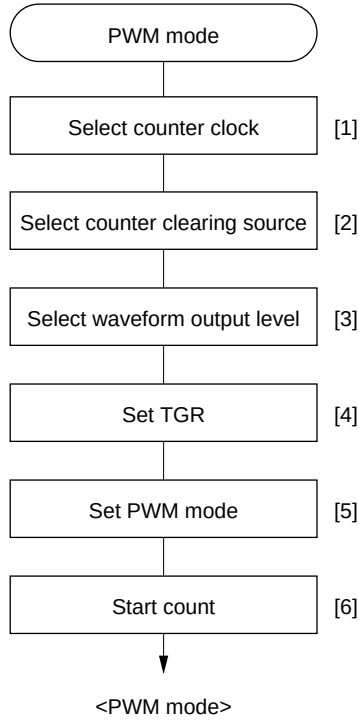
The correspondence between PWM output pins and registers is shown in table 9.7.

**Table 9.7 PWM Output Registers and Output Pins**

| Channel | Registers | Output Pins | |
|---|---|---|---|
| | | PWM Mode 1 | PWM Mode 2 |
| 0 | TGR0A | TIOCA0 | TIOCA0 |
| | TGR0B | | TIOCB0 |
| | TGR0C | TIOCC0 | TIOCC0 |
| | TGR0D | | TIOCD0 |
| 1 | TGR1A | TIOCA1 | TIOCA1 |
| | TGR1B | | TIOCB1 |
| 2 | TGR2A | TIOCA2 | TIOCA2 |
| | TGR2B | | TIOCB2 |
| 3 | TGR3A | TIOCA3 | TIOCA3 |
| | TGR3B | | TIOCB3 |
| | TGR3C | TIOCC3 | TIOCC3 |
| | TGR3D | | TIOCD3 |
| 4 | TGR4A | TIOCA4 | TIOCA4 |
| | TGR4B | | TIOCB4 |
| 5 | TGR5A | TIOCA5 | TIOCA5 |
| | TGR5B | | TIOCB5 |

Note: In PWM mode 2, PWM output is not possible for the TGR register in which the period is set.

**Example of PWM Mode Setting Procedure:** Figure 9.24 shows an example of the PWM mode setting procedure.




[1] Select the counter clock with bits TPSC2 to TPSC0 in TCR. At the same time, select the input clock edge with bits CKEG1 and CKEG0 in TCR.

[2] Use bits CCLR2 to CCLR0 in TCR to select the TGR to be used as the TCNT clearing source.

[3] Use TIOR to designate the TGR as an output compare register, and select the initial value and output value.

[4] Set the cycle in the TGR selected in [2], and set the duty in the other the TGR.

[5] Select the PWM mode with bits MD3 to MD0 in TMDR.

[6] Set the CST bit in TSTR to 1 to start the count operation.

**Figure 9.24 Example of PWM Mode Setting Procedure**

**Examples of PWM Mode Operation:** Figure 9.25 shows an example of PWM mode 1 operation.

In this example, TGRA compare match is set as the TCNT clearing source, 0 is set for the TGRA initial output value and output value, and 1 is set as the TGRB output value.

In this case, the value set in TGRA is used as the period, and the values set in TGRB registers as the duty.

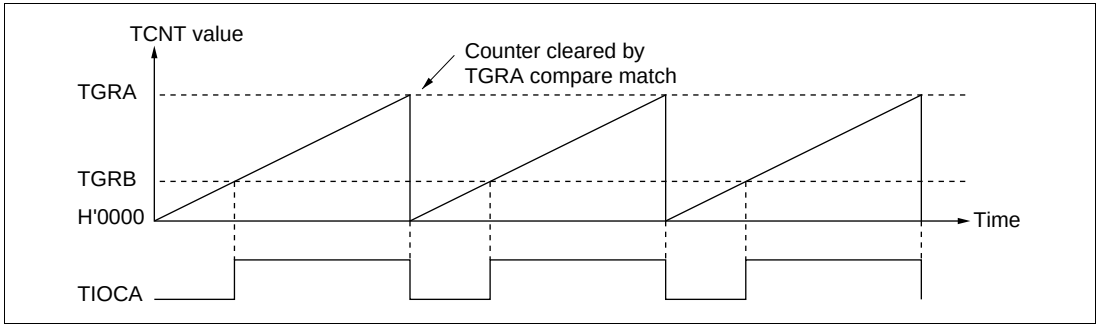



**Figure 9.25 Example of PWM Mode Operation (1)**

Figure 9.26 shows an example of PWM mode 2 operation.

In this example, synchronous operation is designated for channels 0 and 1, TGR1B compare match is set as the TCNT clearing source, and 0 is set for the initial output value and 1 for the output value of the other TGR registers (TGR0A to TGR0D, TGR1A), to output a 5-phase PWM waveform.

In this case, the value set in TGR1B is used as the cycle, and the values set in the other TGRs as the duty.

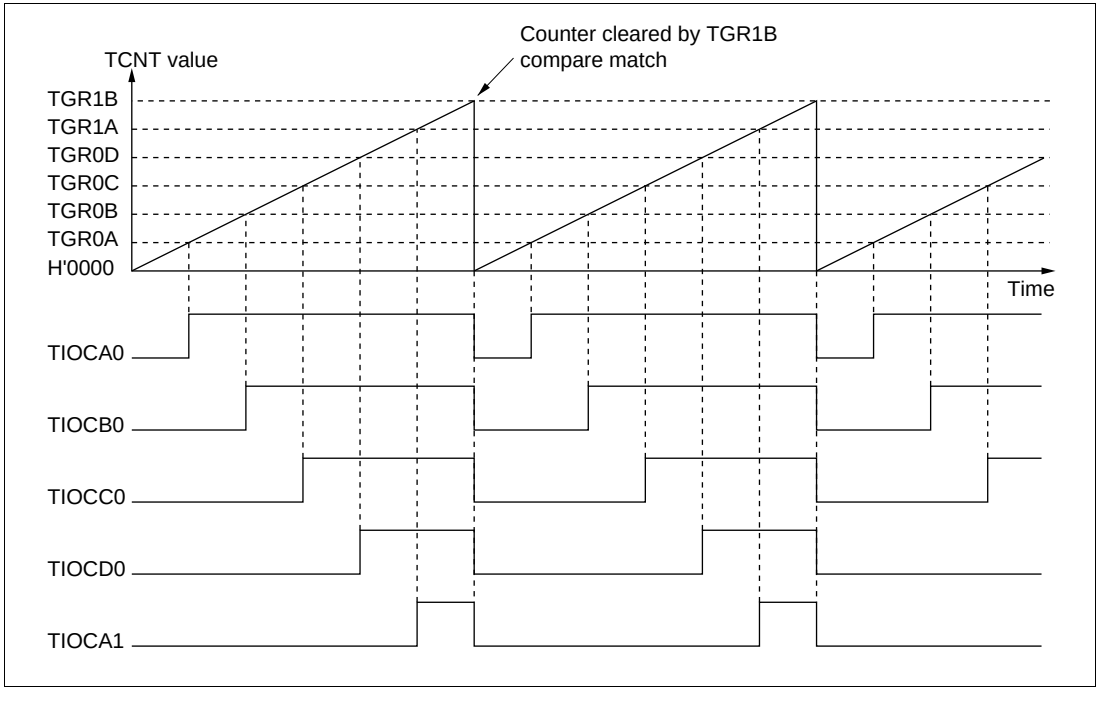



**Figure 9.26 Example of PWM Mode Operation (2)**

Figure 9.27 shows examples of PWM waveform output with 0% duty and 100% duty in PWM mode.

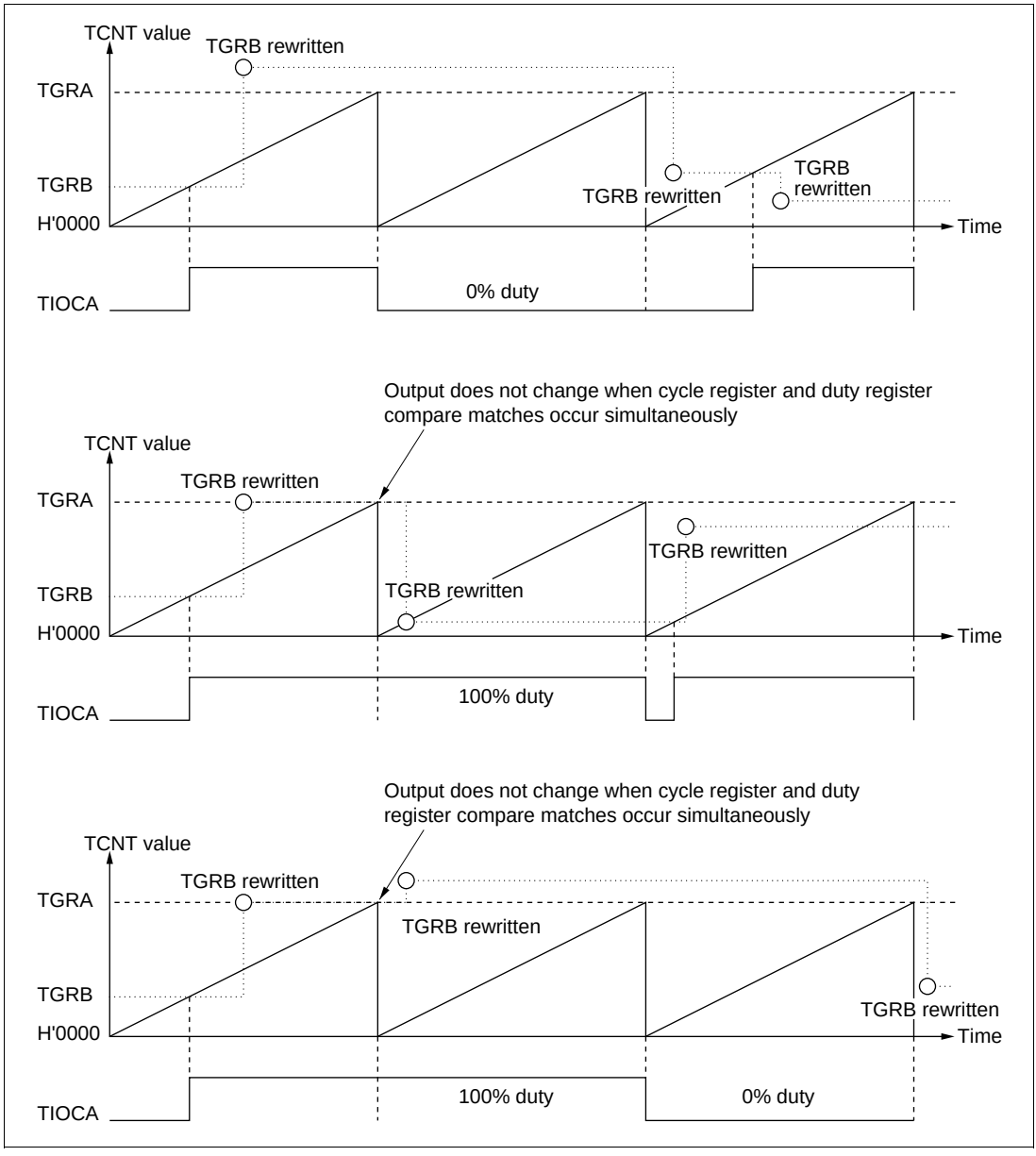



**Figure 9.27 Example of PWM Mode Operation (3)**

### 9.4.7 Phase Counting Mode

In phase counting mode, the phase difference between two external clock inputs is detected and TCNT is incremented/decremented accordingly. This mode can be set for channels 1, 2, 4, and 5.

When phase counting mode is set, an external clock is selected as the counter input clock and TCNT operates as an up/down-counter regardless of the setting of bits TPSC2 to TPSC0 and bits CKEG1 and CKEG0 in TCR. However, the functions of bits CCLR1 and CCLR0 in TCR, and of TIOR, TIER, and TGR are valid, and input capture/compare match and interrupt functions can be used.

When overflow occurs while TCNT is counting up, the TCFV flag in TSR is set; when underflow occurs while TCNT is counting down, the TCFU flag is set.

The TCFD bit in TSR is the count direction flag. Reading the TCFD flag provides an indication of whether TCNT is counting up or down.

Table 9.8 shows the correspondence between external clock pins and channels.

**Table 9.8 Phase Counting Mode Clock Input Pins**

| | External Clock Pins | |
|---|---|---|
| **Channels** | **A-Phase** | **B-Phase** |
| When channel 1 or 5 is set to phase counting mode | TCLKA | TCLKB |
| When channel 2 or 4 is set to phase counting mode | TCLKC | TCLKD |

**Example of Phase Counting Mode Setting Procedure:** Figure 9.28 shows an example of the phase counting mode setting procedure.

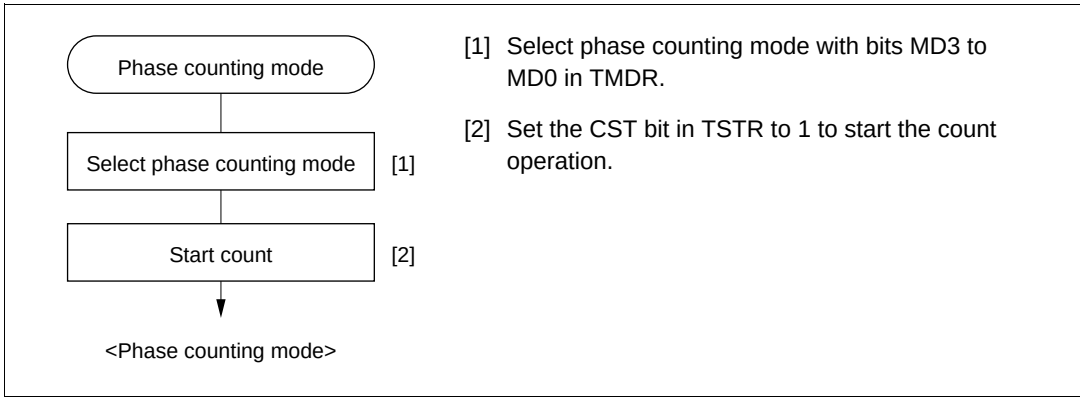



**Figure 9.28 Example of Phase Counting Mode Setting Procedure**

**Examples of Phase Counting Mode Operation:** In phase counting mode, TCNT counts up or down according to the phase difference between two external clocks. There are four modes, according to the count conditions.

- Phase counting mode 1

  Figure 9.29 shows an example of phase counting mode 1 operation, and table 9.9 summarizes the TCNT up/down-count conditions.

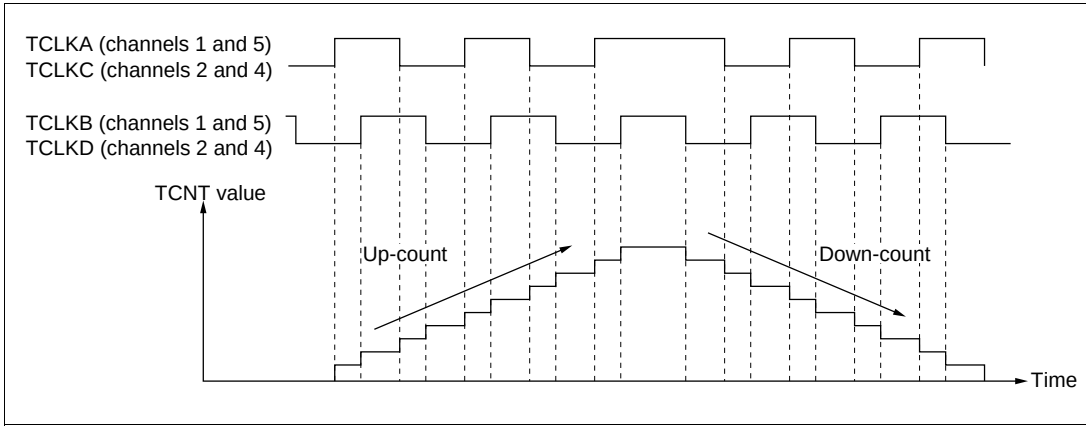



**Figure 9.29 Example of Phase Counting Mode 1 Operation**

**Table 9.9 Up/Down-Count Conditions in Phase Counting Mode 1**

| TCLKA (Channels 1 and 5) TCLKC (Channels 2 and 4) | TCLKB (Channels 1 and 5) TCLKD (Channels 2 and 4) | Operation |
|---|---|---|
| High level | ↑ | Up-count |
| Low level | ↓ | |
| ↑ | Low level | |
| ↓ | High level | |
| High level | ↓ | Down-count |
| Low level | ↑ | |
| ↑ | High level | |
| ↓ | Low level | |

Legend

↑ : Rising edge

↓ : Falling edge

- Phase counting mode 2

  Figure 9.30 shows an example of phase counting mode 2 operation, and table 9.10 summarizes the TCNT up/down-count conditions.

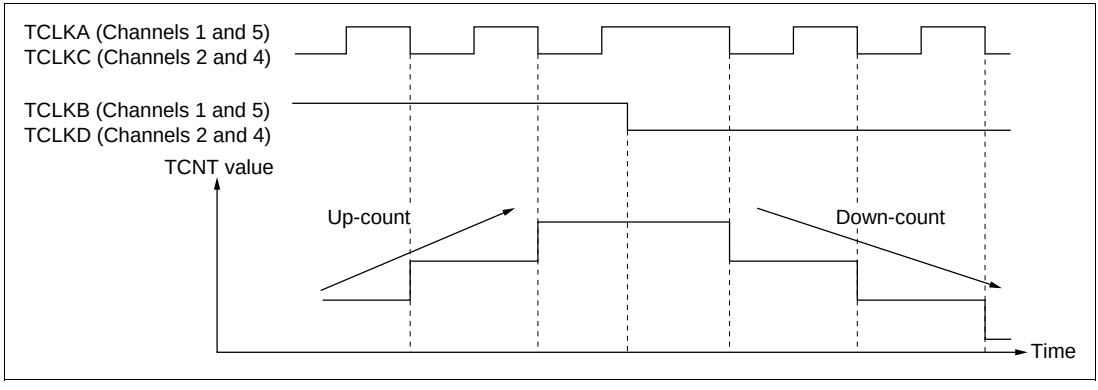



**Figure 9.30 Example of Phase Counting Mode 2 Operation**

**Table 9.10 Up/Down-Count Conditions in Phase Counting Mode 2**

| TCLKA (Channels 1 and 5)<br>TCLKC (Channels 2 and 4) | TCLKB (Channels 1 and 5)<br>TCLKD (Channels 2 and 4) | Operation |
|---|---|---|
| High level | ↑ | Don't care |
| Low level | ↓ | Don't care |
| ↑ | Low level | Don't care |
| ↓ | High level | Up-count |
| High level | ↓ | Don't care |
| Low level | ↑ | Don't care |
| ↑ | High level | Don't care |
| ↓ | Low level | Down-count |

Legend

↑ : Rising edge

↓ : Falling edge

- Phase counting mode 3

  Figure 9.31 shows an example of phase counting mode 3 operation, and table 9.11 summarizes the TCNT up/down-count conditions.

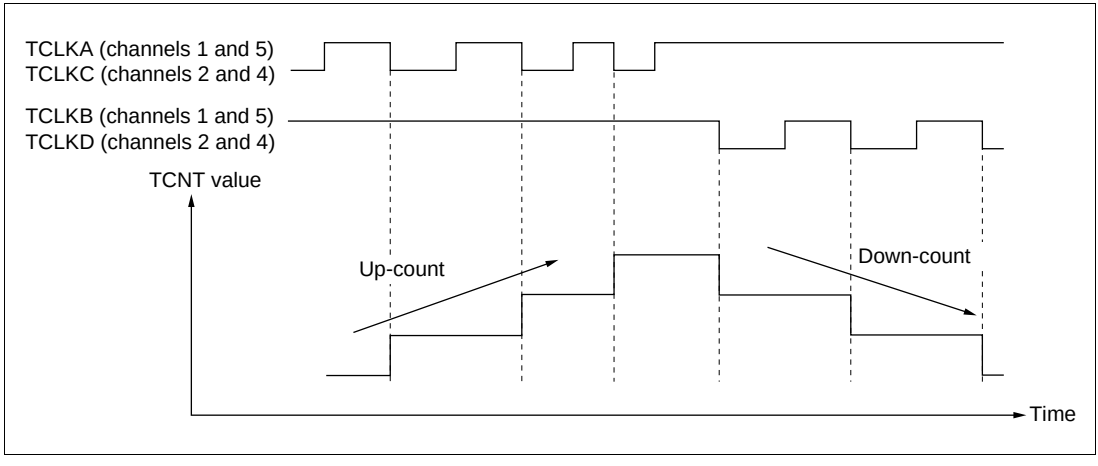



**Figure 9.31 Example of Phase Counting Mode 3 Operation**

**Table 9.11 Up/Down-Count Conditions in Phase Counting Mode 3**

| TCLKA (Channels 1 and 5)<br>TCLKC (Channels 2 and 4) | TCLKB (Channels 1 and 5)<br>TCLKD (Channels 2 and 4) | Operation |
|---|---|---|
| High level | ↑ | Don't care |
| Low level | ↓ | Don't care |
| ↑ | Low level | Don't care |
| ↓ | High level | Up-count |
| High level | ↓ | Down-count |
| Low level | ↑ | Don't care |
| ↑ | High level | Don't care |
| ↓ | Low level | Don't care |

Legend

↑ : Rising edge

↓ : Falling edge

- Phase counting mode 4

  Figure 9.32 shows an example of phase counting mode 4 operation, and table 9.12 summarizes the TCNT up/down-count conditions.

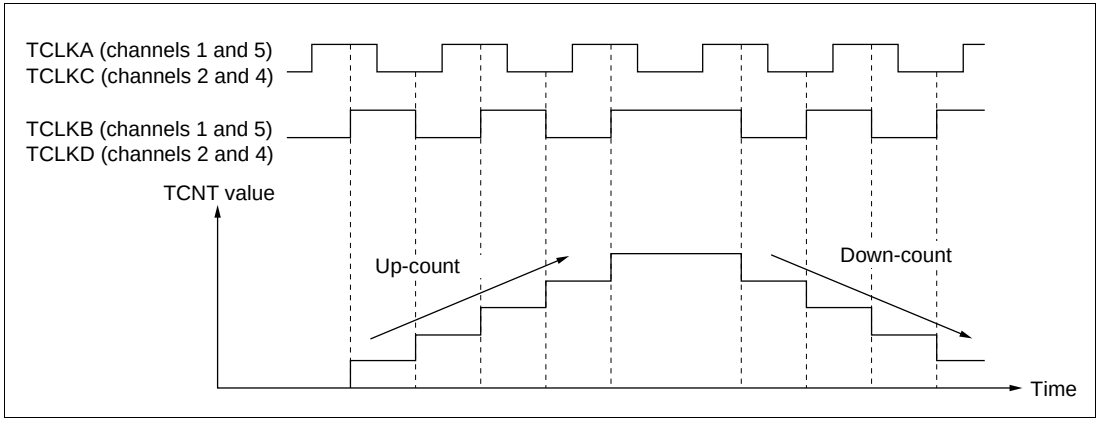



**Figure 9.32 Example of Phase Counting Mode 4 Operation**

**Table 9.12 Up/Down-Count Conditions in Phase Counting Mode 4**

| TCLKA (Channels 1 and 5) TCLKC (Channels 2 and 4) | TCLKB (Channels 1 and 5) TCLKD (Channels 2 and 4) | Operation |
|---|---|---|
| High level | ↑ | Up-count |
| Low level | ↓ | |
| ↑ | Low level | Don't care |
| ↓ | High level | |
| High level | ↓ | Down-count |
| Low level | ↑ | |
| ↑ | High level | Don't care |
| ↓ | Low level | |

Legend

↑ : Rising edge

↓ : Falling edge

**Phase Counting Mode Application Example:** Figure 9.33 shows an example in which phase counting mode is designated for channel 1, and channel 1 is coupled with channel 0 to input servo motor 2-phase encoder pulses in order to detect the position or speed.

Channel 1 is set to phase counting mode 1, and the encoder pulse A-phase and B-phase are input to TCLKA and TCLKB.

Channel 0 operates with TCNT counter clearing by TGR0C compare match; TGR0A and TGR0C are used for the compare match function, and are set with the speed control period and position control period. TGR0B is used for input capture, with TGR0B and TGR0D operating in buffer mode. The channel 1 counter input clock is designated as the TGR0B input capture source, and detection of the pulse width of 2-phase encoder 4-multiplication pulses is performed.

TGR1A and TGR1B for channel 1 are designated for input capture, channel 0 TGR0A and TGR0C compare matches are selected as the input capture source, and store the up/down-counter values for the control periods.

This procedure enables accurate position/speed detection to be achieved.

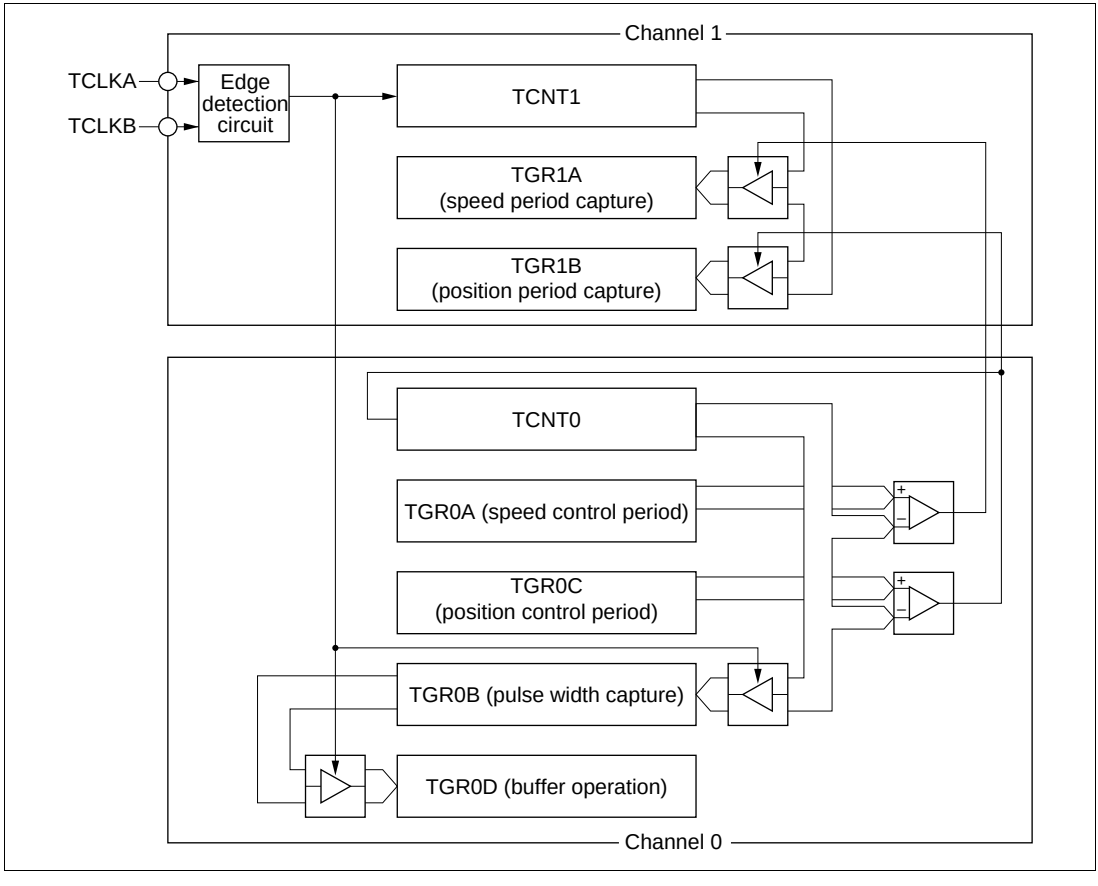



**Figure 9.33 Phase Counting Mode Application Example**

## 9.5 Interrupts

### 9.5.1 Interrupt Sources and Priorities

There are three kinds of TPU interrupt source: TGR input capture/compare match, TCNT overflow, and TCNT underflow. Each interrupt source has its own status flag and enable/disabled bit, allowing generation of interrupt request signals to be enabled or disabled individually.

When an interrupt request is generated, the corresponding status flag in TSR is set to 1. If the corresponding enable/disable bit in TIER is set to 1 at this time, an interrupt is requested. The interrupt request is cleared by clearing the status flag to 0.

Relative channel priorities can be changed by the interrupt controller, but the priority order within a channel is fixed. For details, see section 5, Interrupt Controller.

Table 9.13 lists the TPU interrupt sources.

**Table 9.13 TPU Interrupts**

| Channel | Interrupt Source | Description | DTC Activation | Priority |
|---|---|---|---|---|
| 0 | TGI0A | TGR0A input capture/compare match | Possible | High |
| | TGI0B | TGR0B input capture/compare match | Possible | |
| | TGI0C | TGR0C input capture/compare match | Possible | |
| | TGI0D | TGR0D input capture/compare match | Possible | |
| | TCI0V | TCNT0 overflow | Not possible | |
| 1 | TGI1A | TGR1A input capture/compare match | Possible | |
| | TGI1B | TGR1B input capture/compare match | Possible | |
| | TCI1V | TCNT1 overflow | Not possible | |
| | TCI1U | TCNT1 underflow | Not possible | |
| 2 | TGI2A | TGR2A input capture/compare match | Possible | |
| | TGI2B | TGR2B input capture/compare match | Possible | |
| | TCI2V | TCNT2 overflow | Not possible | |
| | TCI2U | TCNT2 underflow | Not possible | |
| 3 | TGI3A | TGR3A input capture/compare match | Possible | |
| | TGI3B | TGR3B input capture/compare match | Possible | |
| | TGI3C | TGR3C input capture/compare match | Possible | |
| | TGI3D | TGR3D input capture/compare match | Possible | |
| | TCI3V | TCNT3 overflow | Not possible | |
| 4 | TGI4A | TGR4A input capture/compare match | Possible | |
| | TGI4B | TGR4B input capture/compare match | Possible | |
| | TCI4V | TCNT4 overflow | Not possible | |
| | TCI4U | TCNT4 underflow | Not possible | |
| 5 | TGI5A | TGR5A input capture/compare match | Possible | |
| | TGI5B | TGR5B input capture/compare match | Possible | |
| | TCI5V | TCNT5 overflow | Not possible | |
| | TCI5U | TCNT5 underflow | Not possible | Low |

Note: This table shows the initial state immediately after a reset. The relative channel priorities can be changed by the interrupt controller.

**Input Capture/Compare Match Interrupt:** An interrupt is requested if the TGIE bit in TIER is set to 1 when the TGF flag in TSR is set to 1 by the occurrence of a TGR input capture/compare match on a particular channel. The interrupt request is cleared by clearing the TGF flag to 0. The TPU has 16 input capture/compare match interrupts, four each for channels 0 and 3, and two each for channels 1, 2, 4, and 5.

**Overflow Interrupt:** An interrupt is requested if the TCIEV bit in TIER is set to 1 when the TCFV flag in TSR is set to 1 by the occurrence of TCNT overflow on a channel. The interrupt request is cleared by clearing the TCFV flag to 0. The TPU has six overflow interrupts, one for each channel.

**Underflow Interrupt:** An interrupt is requested if the TCIEU bit in TIER is set to 1 when the TCFU flag in TSR is set to 1 by the occurrence of TCNT underflow on a channel. The interrupt request is cleared by clearing the TCFU flag to 0. The TPU has four overflow interrupts, one each for channels 1, 2, 4, and 5.

### 9.5.2 DTC Activation

**DTC Activation:** The DTC can be activated by the TGR input capture/compare match interrupt for a channel. For details, see section 7, Data Transfer Controller.

A total of 16 TPU input capture/compare match interrupts can be used as DTC activation sources, four each for channels 0 and 3, and two each for channels 1, 2, 4, and 5.

### 9.5.3 A/D Converter Activation

The A/D converter can be activated by the TGRA input capture/compare match for a channel.

If the TTGE bit in TIER is set to 1 when the TGFA flag in TSR is set to 1 by the occurrence of a TGRA input capture/compare match on a particular channel, a request to start A/D conversion is sent to the A/D converter. If the TPU conversion start trigger has been selected on the A/D converter side at this time, A/D conversion is started.

In the TPU, a total of six TGRA input capture/compare match interrupts can be used as A/D converter conversion start sources, one for each channel.

## 9.6 Operation Timing

### 9.6.1 Input/Output Timing

**TCNT Count Timing:** Figure 9.34 shows TCNT count timing in internal clock operation, and figure 9.35 shows TCNT count timing in external clock operation.

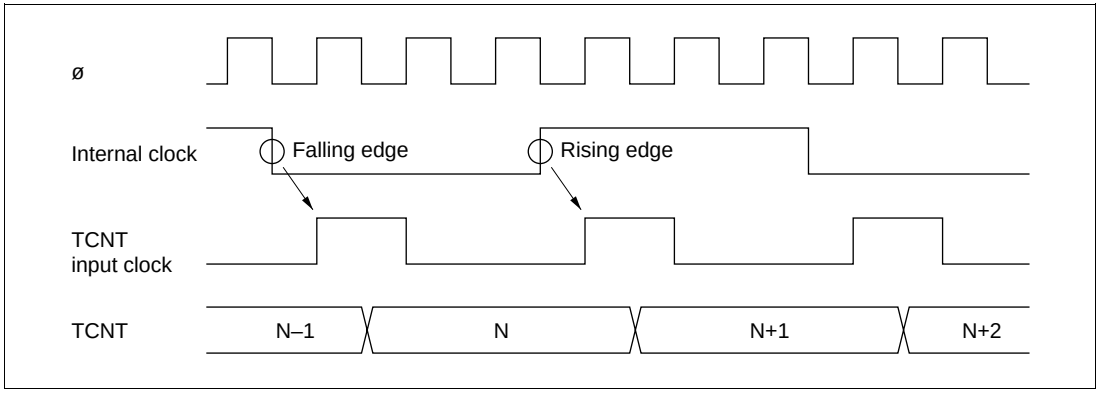



**Figure 9.34 Count Timing in Internal Clock Operation**

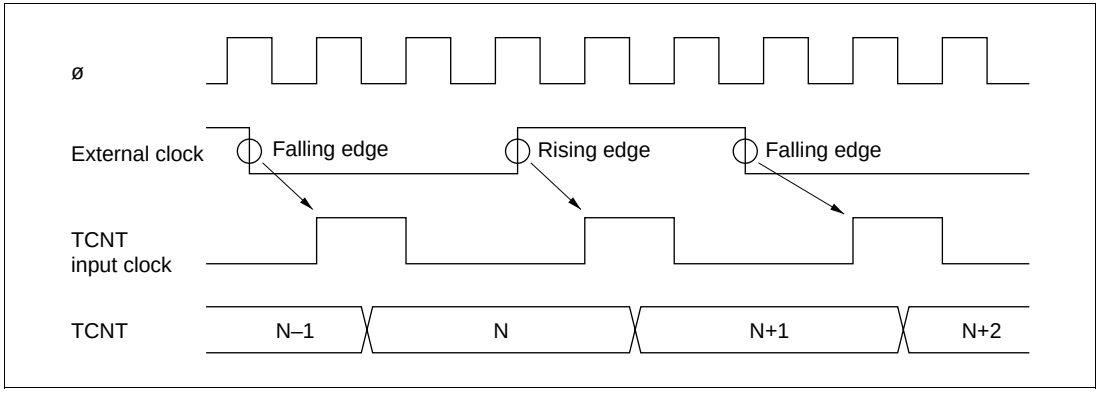



**Figure 9.35 Count Timing in External Clock Operation**

**Output Compare Output Timing:** A compare match signal is generated in the final state in which TCNT and TGR match (the point at which the count value matched by TCNT is updated). When a compare match signal is generated, the output value set in TIOR is output at the output compare output pin. After a match between TCNT and TGR, the compare match signal is not generated until the TCNT input clock is generated.

Figure 9.36 shows output compare output timing.

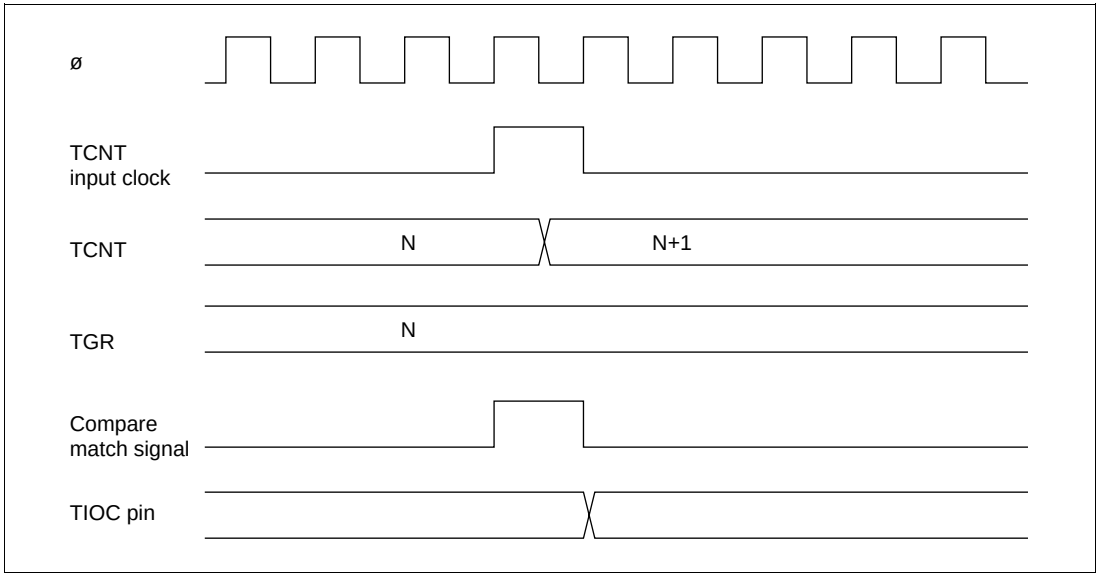



**Figure 9.36 Output Compare Output Timing**

**Input Capture Signal Timing:** Figure 9.37 shows input capture signal timing.

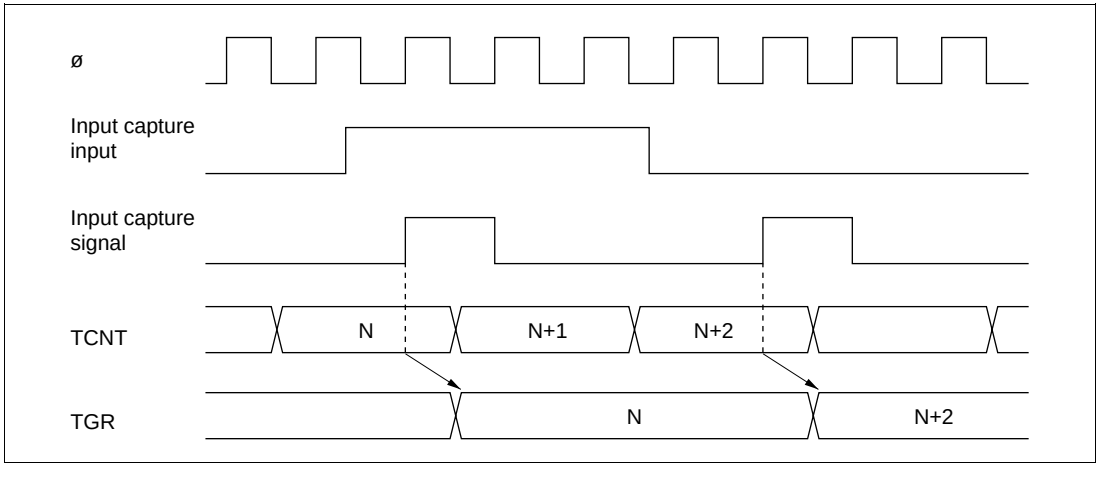



**Figure 9.37 Input Capture Input Signal Timing**

**Timing for Counter Clearing by Compare Match/Input Capture:** Figure 9.38 shows the timing when counter clearing by compare match occurrence is specified, and figure 9.39 shows the timing when counter clearing by input capture occurrence is specified.

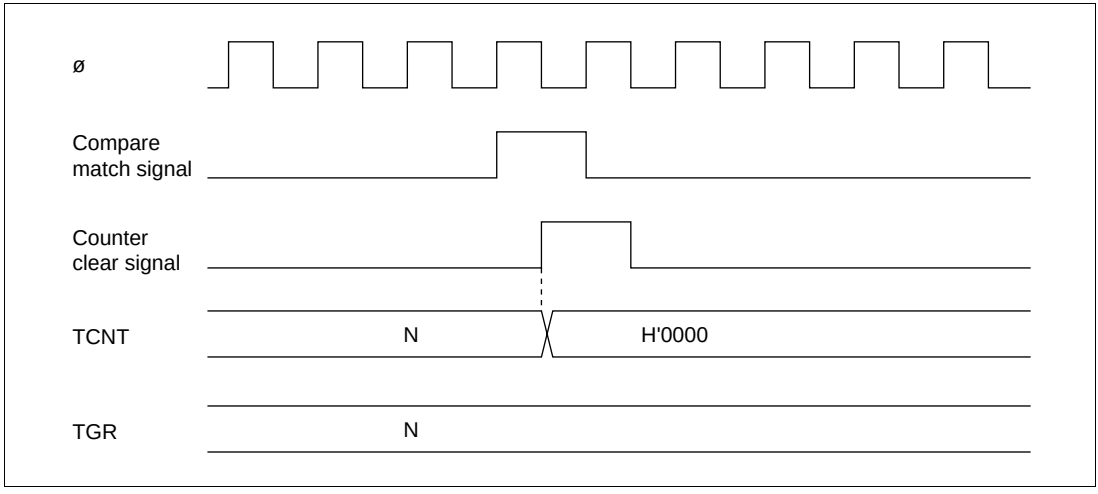



**Figure 9.38 Counter Clear Timing (Compare Match)**

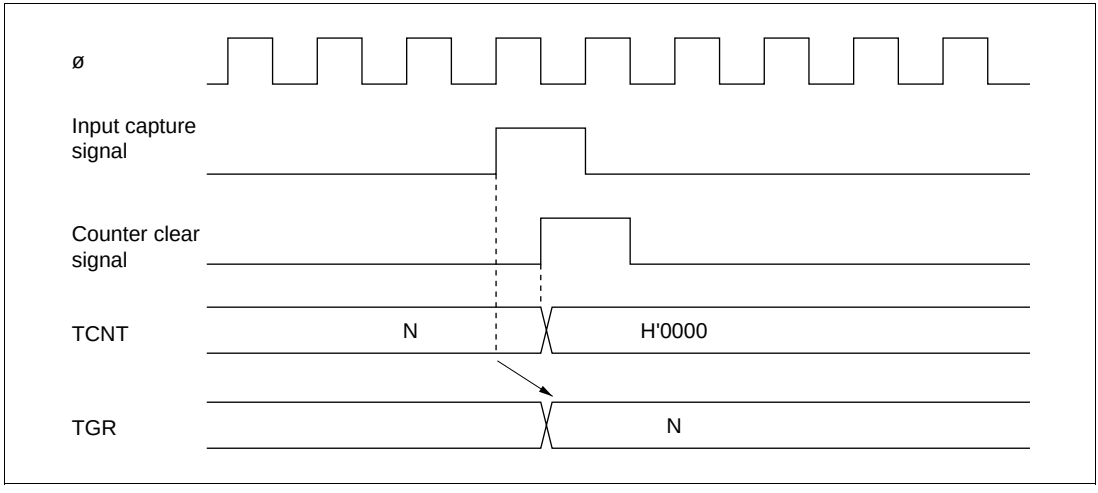



**Figure 9.39 Counter Clear Timing (Input Capture)**

**Buffer Operation Timing:** Figures 9.40 and 9.41 show the timing in buffer operation.

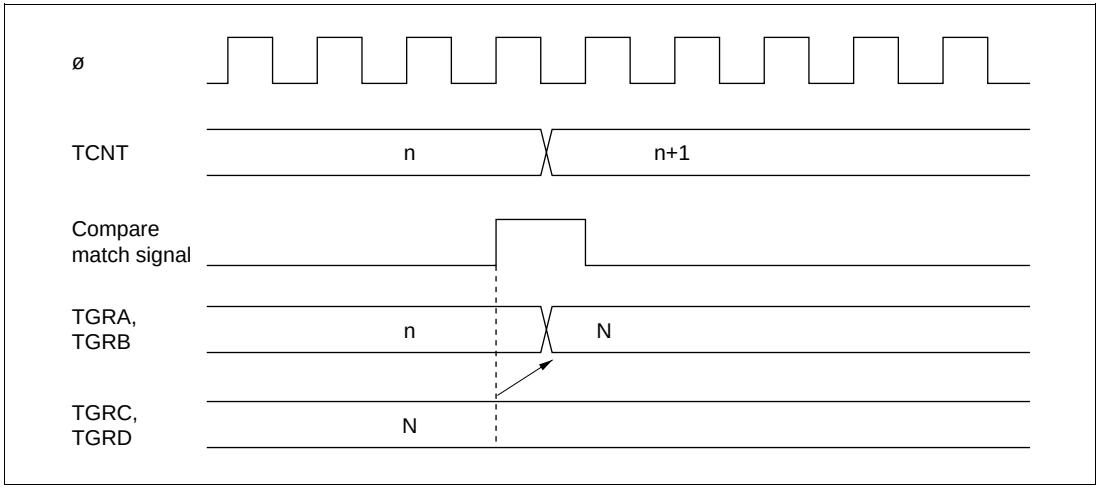



**Figure 9.40 Buffer Operation Timing (Compare Match)**

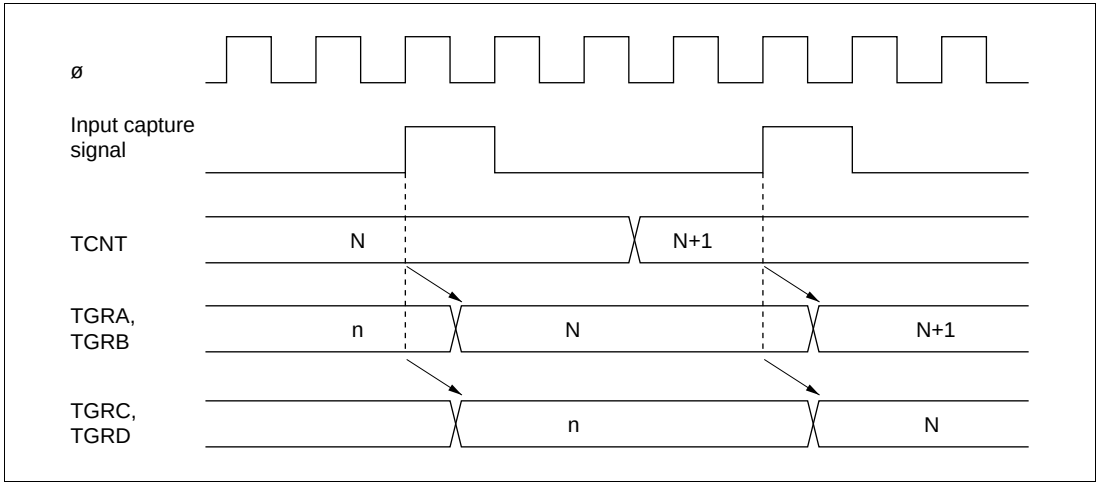



**Figure 9.41 Buffer Operation Timing (Input Capture)**

### 9.6.2 Interrupt Signal Timing

**TGF Flag Setting Timing in Case of Compare Match:** Figure 9.42 shows the timing for setting of the TGF flag in TSR by compare match occurrence, and TGI interrupt request signal timing.

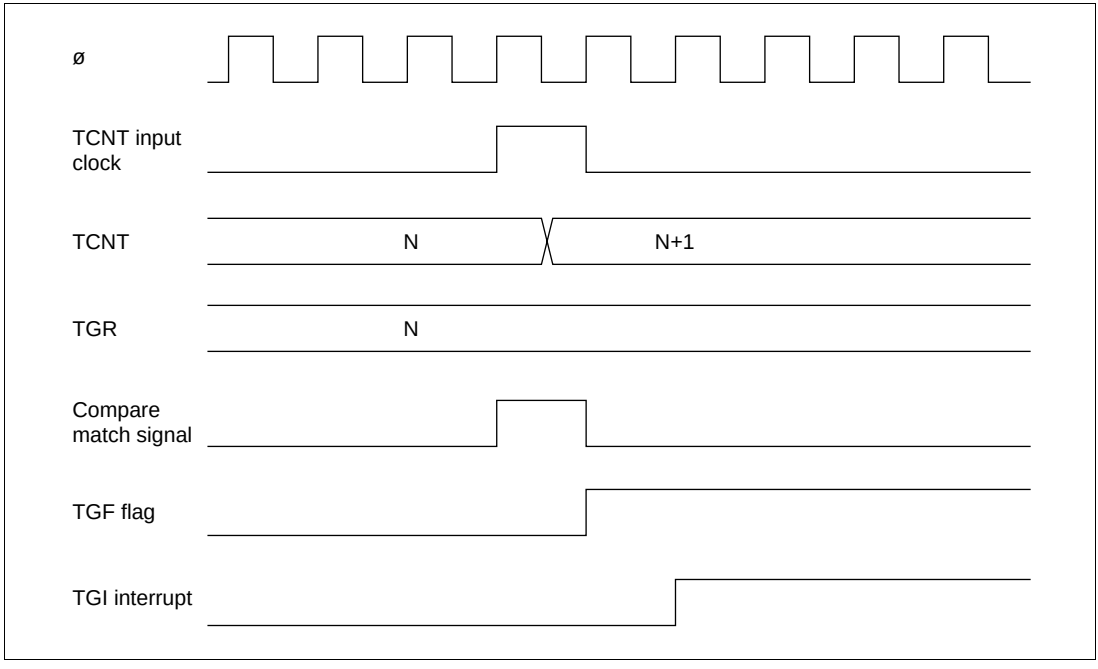



**Figure 9.42 TGI Interrupt Timing (Compare Match)**

**TGF Flag Setting Timing in Case of Input Capture:** Figure 9.43 shows the timing for setting of the TGF flag in TSR by input capture occurrence, and TGI interrupt request signal timing.

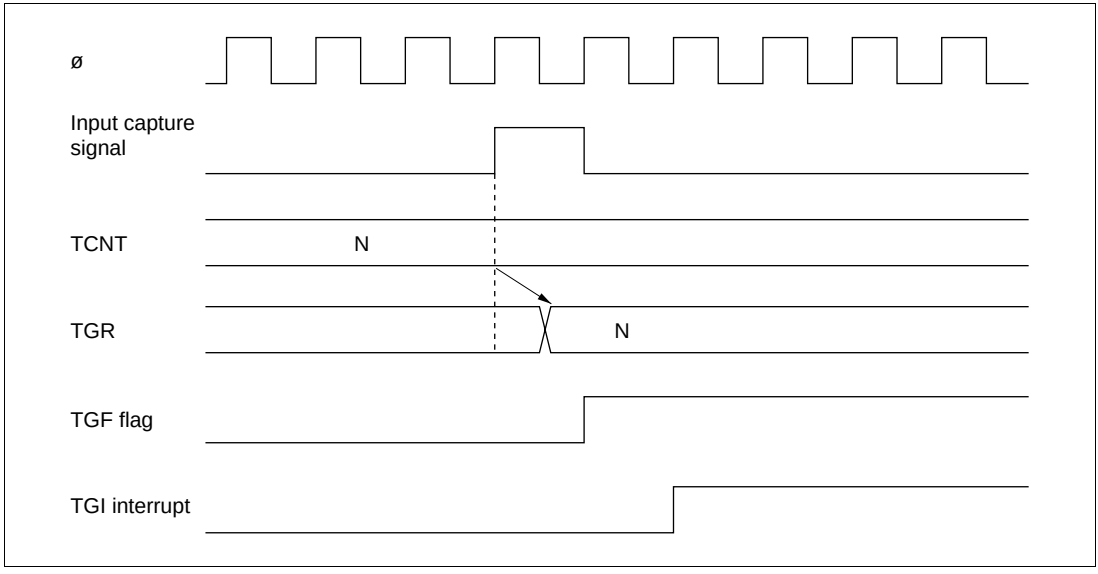



**Figure 9.43 TGI Interrupt Timing (Input Capture)**

**TCFV Flag/TCFU Flag Setting Timing:** Figure 9.44 shows the timing for setting of the TCFV flag in TSR by overflow occurrence, and TCIV interrupt request signal timing.

Figure 9.45 shows the timing for setting of the TCFU flag in TSR by underflow occurrence, and TCIU interrupt request signal timing.

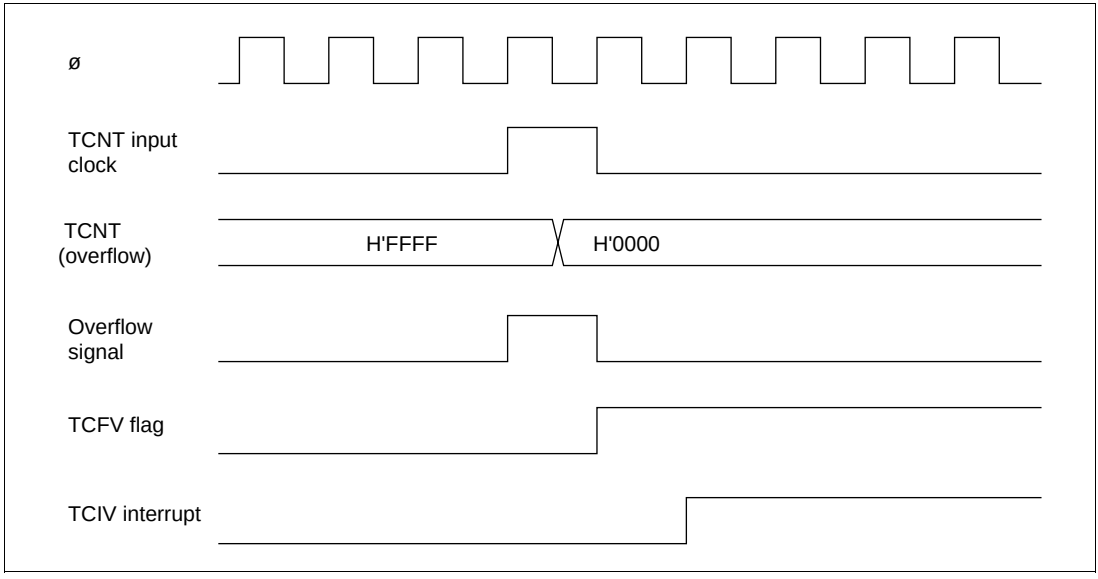



**Figure 9.44 TCIV Interrupt Setting Timing**

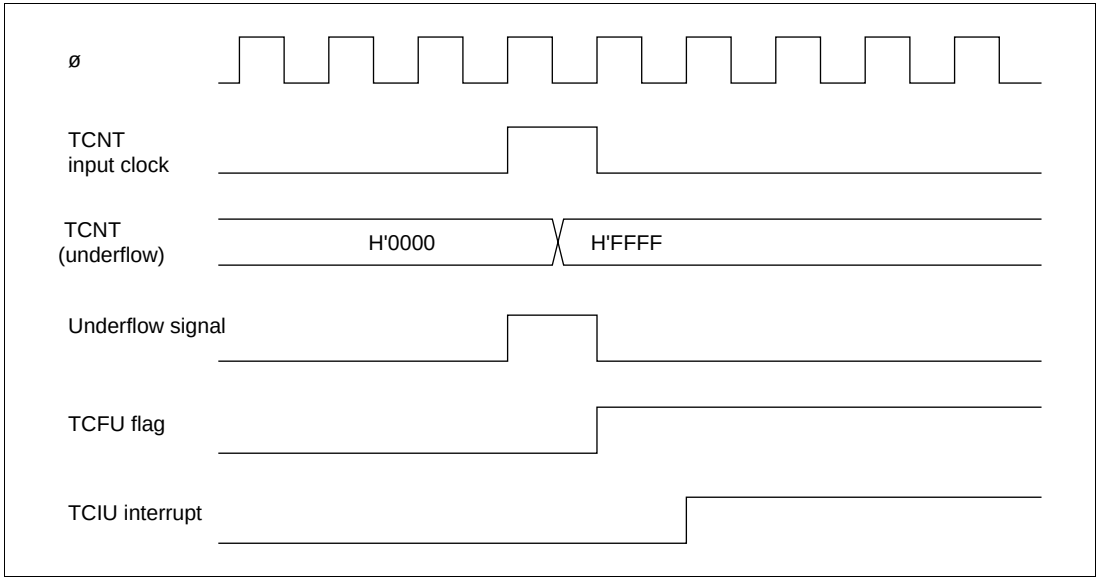



**Figure 9.45 TCIU Interrupt Setting Timing**

**Status Flag Clearing Timing:** After a status flag is read as 1 by the CPU, it is cleared by writing 0 to it. When the DTC is activated, the flag is cleared automatically. Figure 9.46 shows the timing for status flag clearing by the CPU, and figure 9.47 shows the timing for status flag clearing by the DTC.

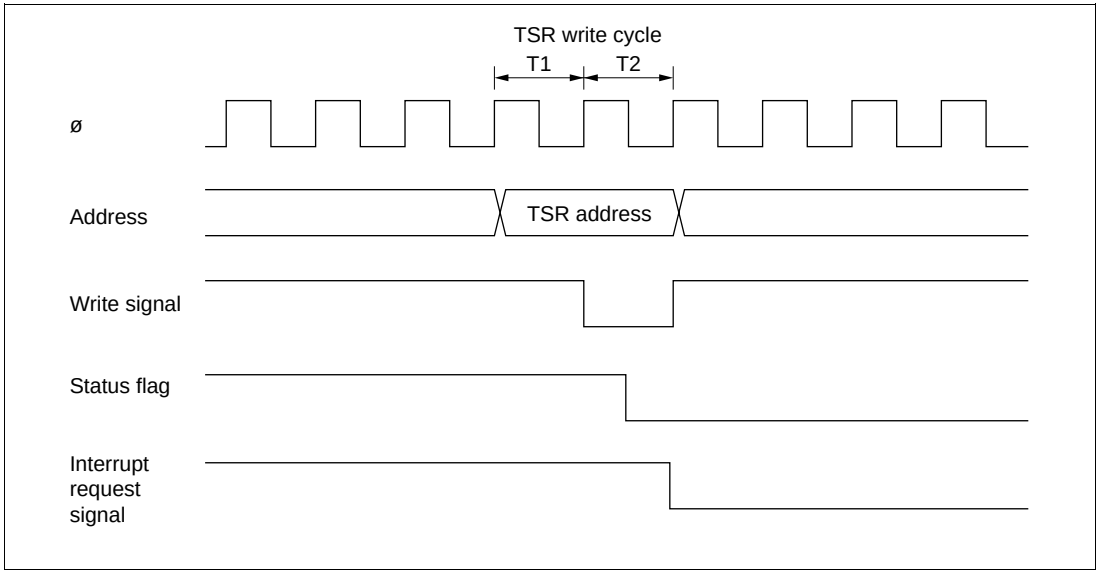



**Figure 9.46 Timing for Status Flag Clearing by CPU**

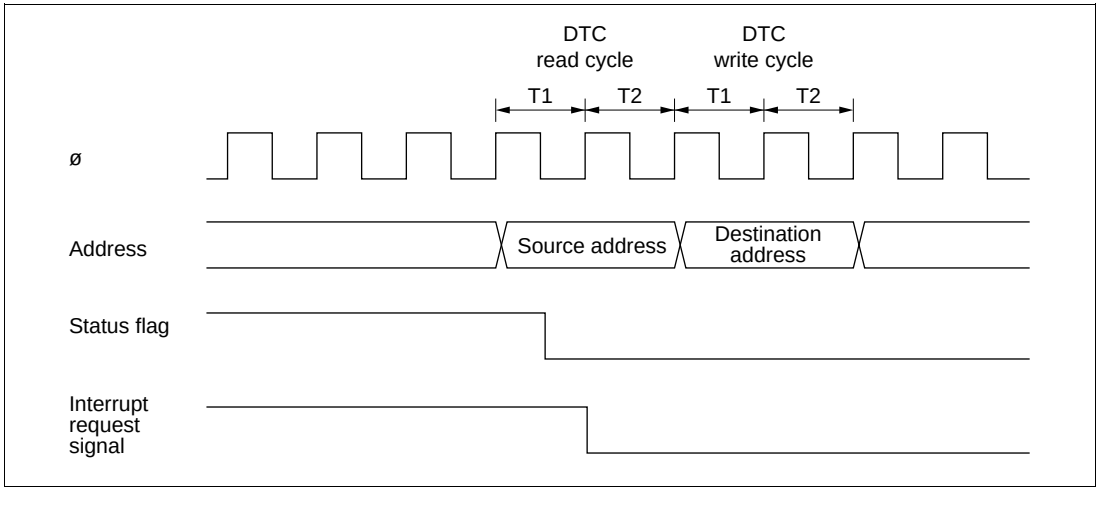



**Figure 9.47 Timing for Status Flag Clearing by DTC Activation**

## 9.7 Usage Notes

Note that the kinds of operation and contention described below occur during TPU operation.

**Input Clock Restrictions:** The input clock pulse width must be at least 1.5 states in the case of single-edge detection, and at least 2.5 states in the case of both-edge detection. The TPU will not operate properly with a narrower pulse width.

In phase counting mode, the phase difference and overlap between the two input clocks must be at least 1.5 states, and the pulse width must be at least 2.5 states. Figure 9.48 shows the input clock conditions in phase counting mode.

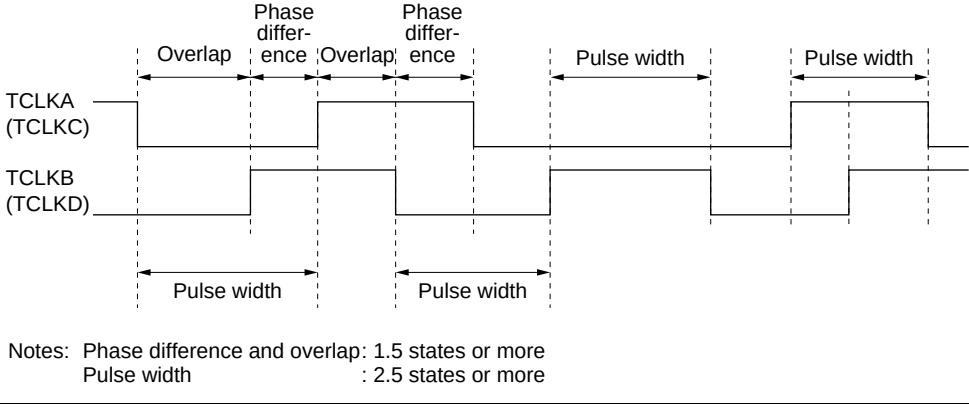



**Figure 9.48 Phase Difference, Overlap, and Pulse Width in Phase Counting Mode**

**Caution on Period Setting:** When counter clearing by compare match is set, TCNT is cleared in the final state in which it matches the TGR value (the point at which the count value matched by TCNT is updated). Consequently, the actual counter frequency is given by the following formula:

$$f = \frac{\phi}{(N + 1)}$$

Where f : Counter frequency
ø : Operating frequency
N : TGR set value

**Contention between TCNT Write and Clear Operations:** If the counter clear signal is generated in the T2 state of a TCNT write cycle, TCNT clearing takes precedence and the TCNT write is not performed.

Figure 9.49 shows the timing in this case.

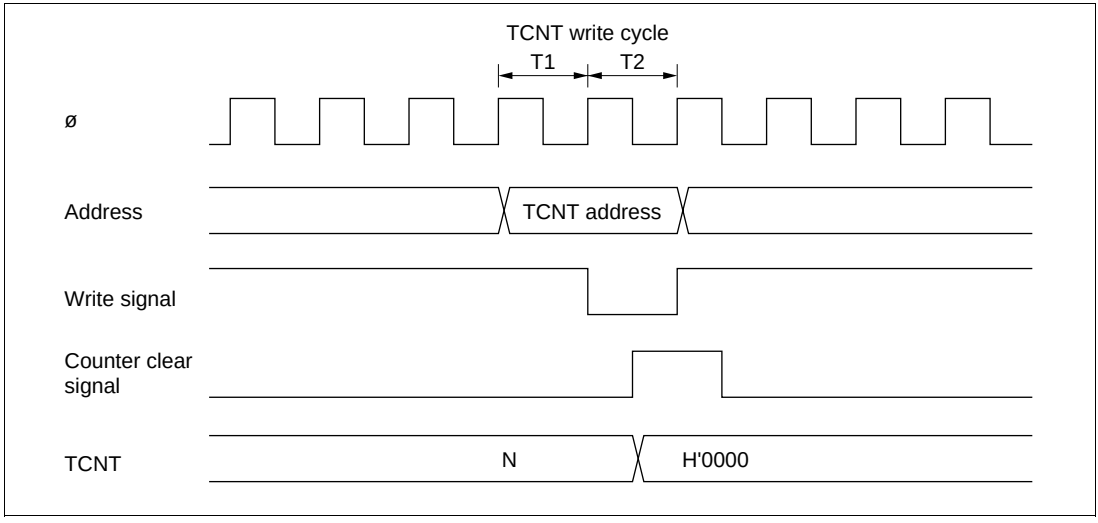



**Figure 9.49 Contention between TCNT Write and Clear Operations**

**Contention between TCNT Write and Increment Operations:** If incrementing occurs in the T2 state of a TCNT write cycle, the TCNT write takes precedence and TCNT is not incremented.

Figure 9.50 shows the timing in this case.

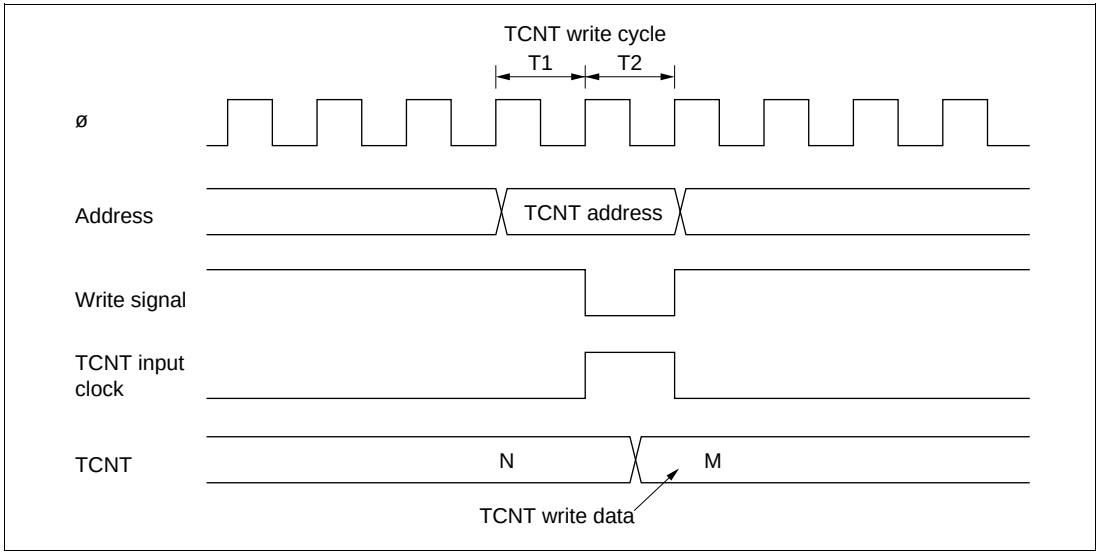



**Figure 9.50 Contention between TCNT Write and Increment Operations**

**Contention between TGR Write and Compare Match:** If a compare match occurs in the T2 state of a TGR write cycle, the TGR write takes precedence and the compare match signal is inhibited. A compare match does not occur even if the same value as before is written.

Figure 9.51 shows the timing in this case.

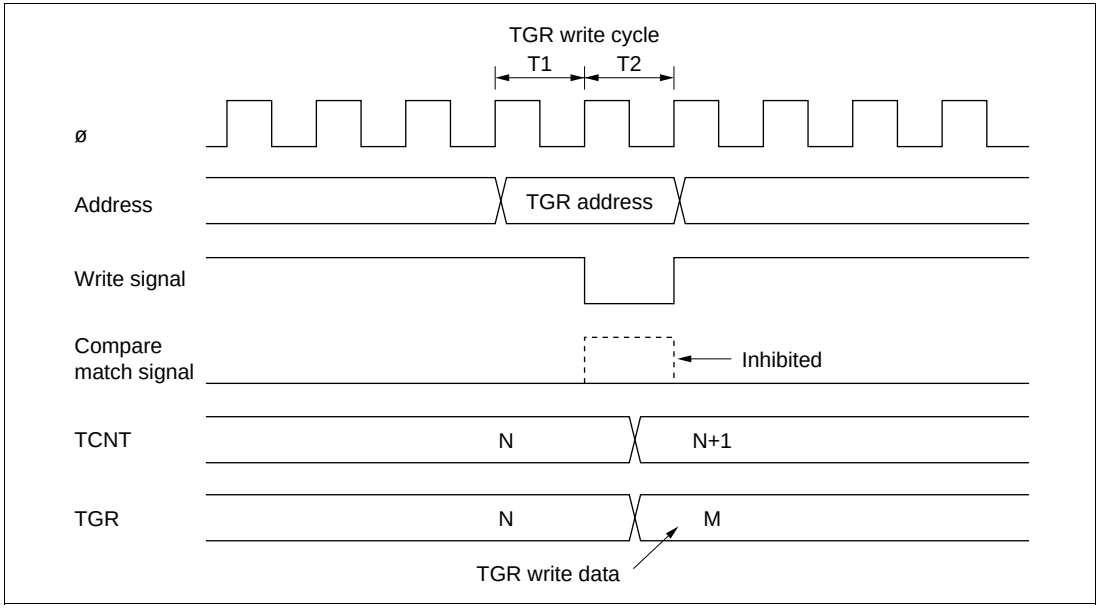



**Figure 9.51 Contention between TGR Write and Compare Match**

**Contention between Buffer Register Write and Compare Match:** If a compare match occurs in the T2 state of a TGR write cycle, the data transferred to TGR by the buffer operation will be the data prior to the write.

Figure 9.52 shows the timing in this case.

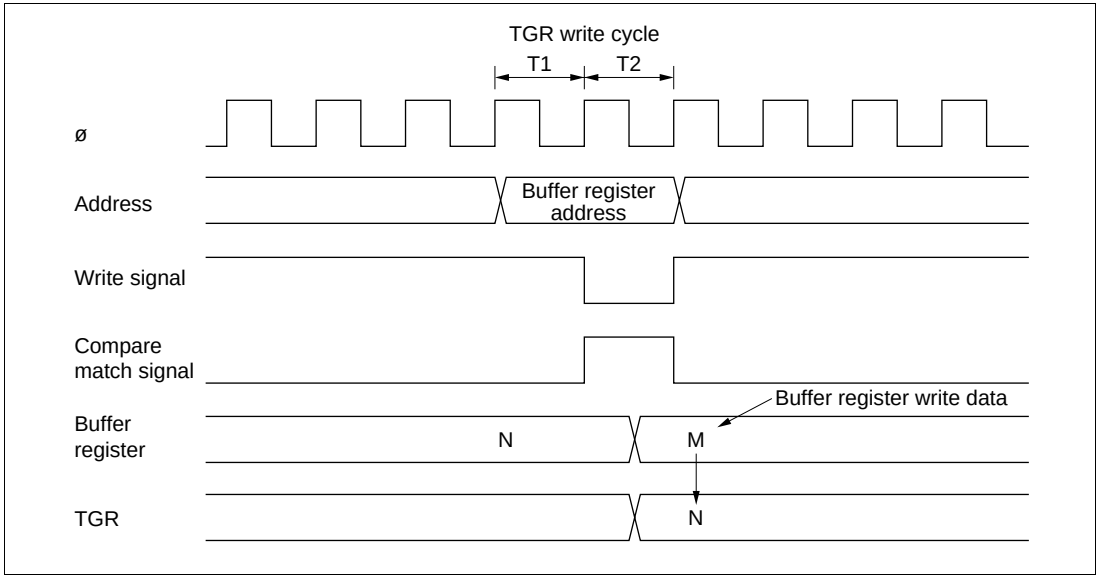



**Figure 9.52 Contention between Buffer Register Write and Compare Match**

**Contention between TGR Read and Input Capture:** If the input capture signal is generated in the T1 state of a TGR read cycle, the data that is read will be the data after input capture transfer.

Figure 9.53 shows the timing in this case.

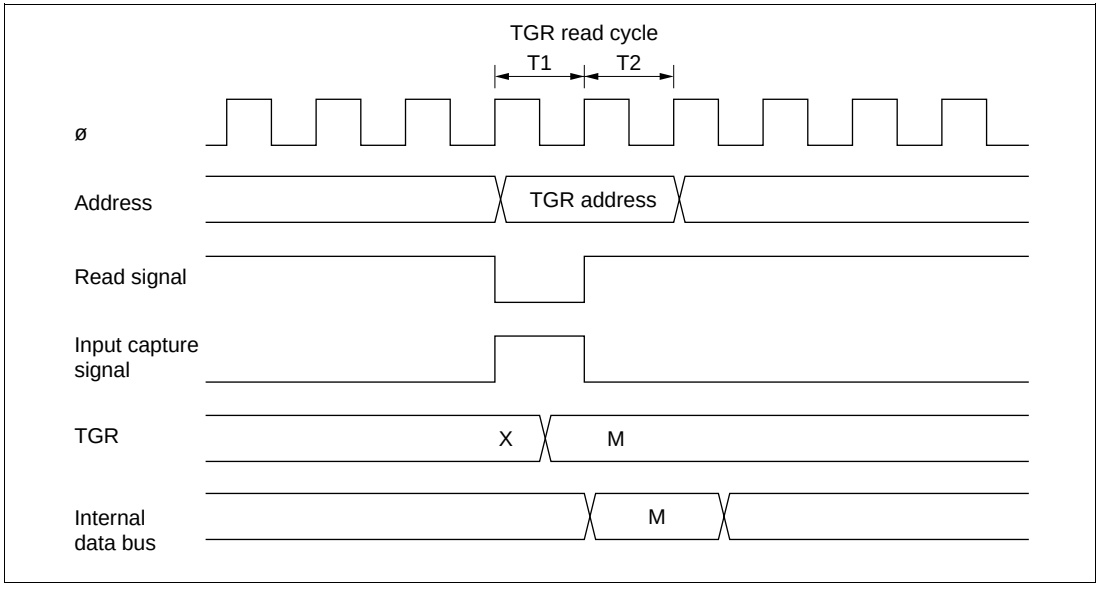



**Figure 9.53 Contention between TGR Read and Input Capture**

**Contention between TGR Write and Input Capture:** If the input capture signal is generated in the T2 state of a TGR write cycle, the input capture operation takes precedence and the write to TGR is not performed.

Figure 9.54 shows the timing in this case.

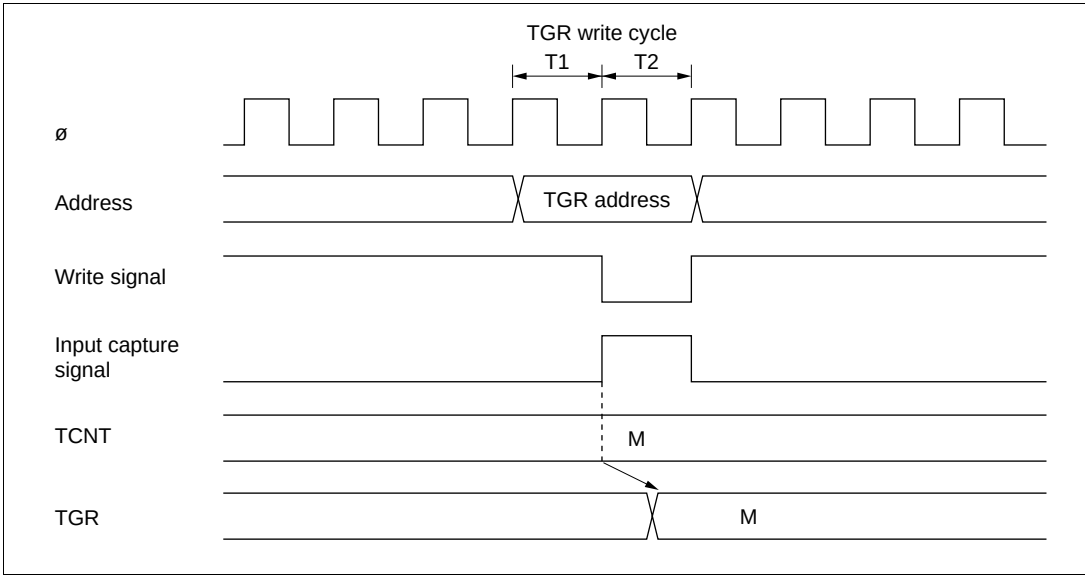



**Figure 9.54 Contention between TGR Write and Input Capture**

**Contention between Buffer Register Write and Input Capture:** If the input capture signal is generated in the T2 state of a buffer write cycle, the buffer operation takes precedence and the write to the buffer register is not performed.

Figure 9.55 shows the timing in this case.

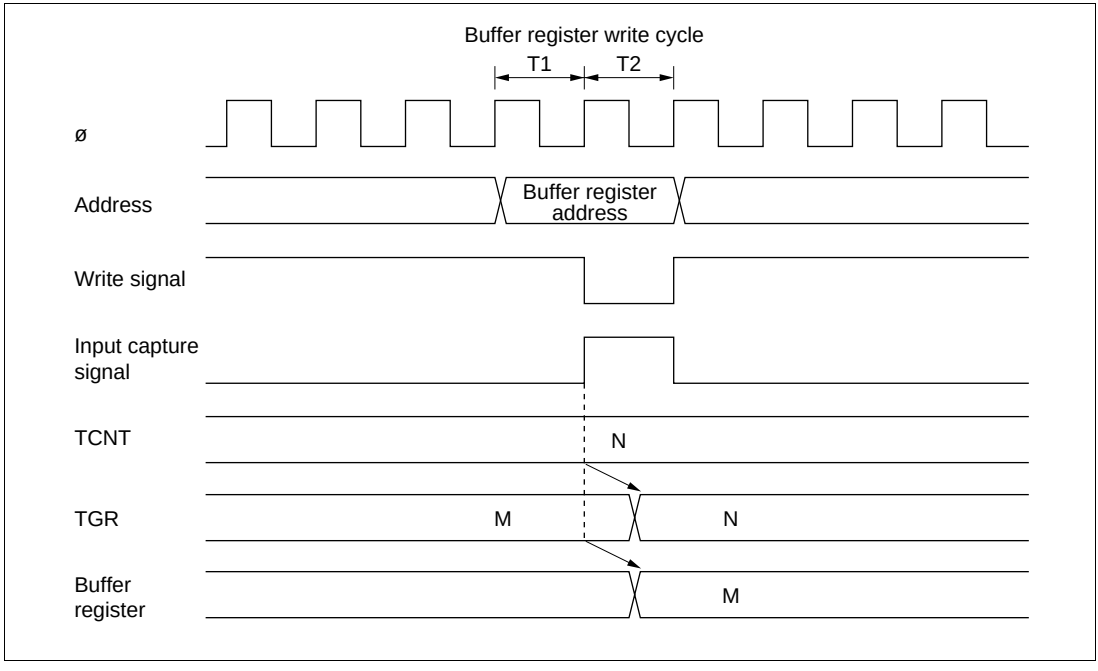



**Figure 9.55 Contention between Buffer Register Write and Input Capture**

**Contention between Overflow/Underflow and Counter Clearing:** If overflow/underflow and counter clearing occur simultaneously, the TCFV/TCFU flag in TSR is not set and TCNT clearing takes precedence.

Figure 9.56 shows the operation timing when a TGR compare match is specified as the clearing source, and H'FFFF is set in TGR.

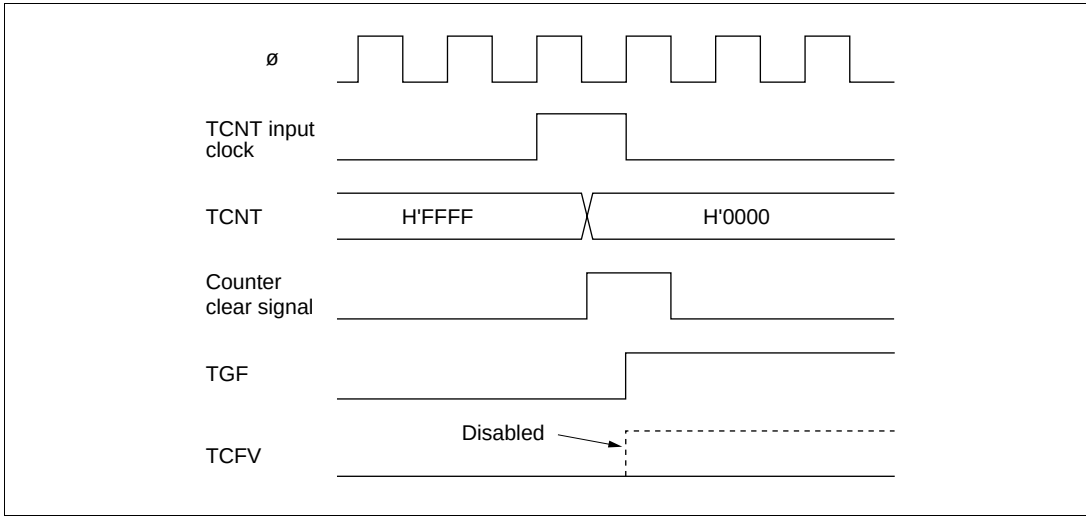



**Figure 9.56 Contention between Overflow and Counter Clearing**

**Contention between TCNT Write and Overflow/Underflow:** If there is an up-count or down-count in the T2 state of a TCNT write cycle, and overflow/underflow occurs, the TCNT write takes precedence and the TCFV/TCFU flag in TSR is not set .

Figure 9.57 shows the operation timing when there is contention between a TCNT write and overflow.

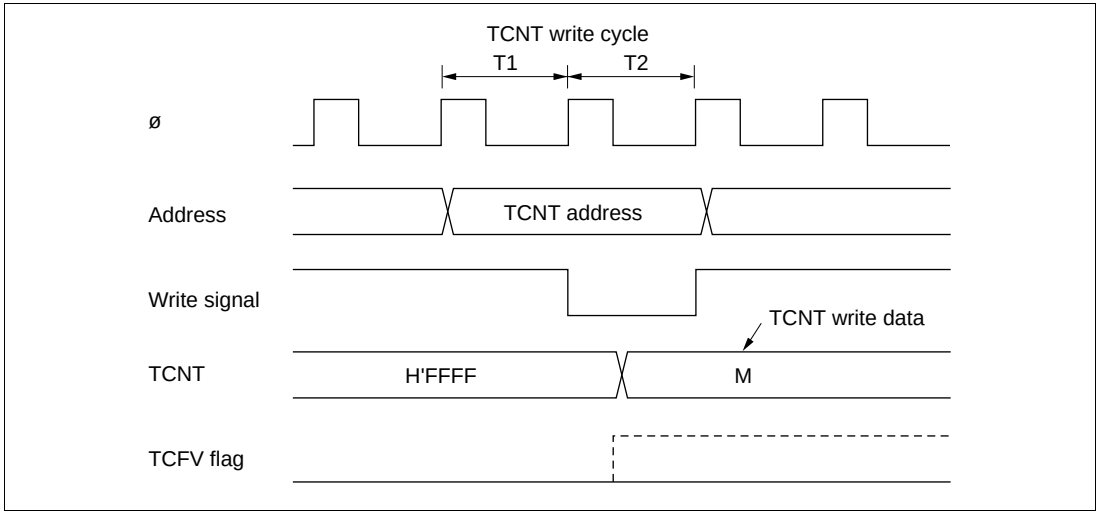



**Figure 9.57 Contention between TCNT Write and Overflow**

**Multiplexing of I/O Pins:** In the H8S/2345 Series, the TCLKA input pin is multiplexed with the TIOCC0 I/O pin, the TCLKB input pin with the TIOCD0 I/O pin, the TCLKC input pin with the TIOCB1 I/O pin, and the TCLKD input pin with the TIOCB2 I/O pin. When an external clock is input, compare match output should not be performed from a multiplexed pin.

**Interrupts and Module Stop Mode:** If module stop mode is entered when an interrupt has been requested, it will not be possible to clear the CPU interrupt source or DTC activation source. Interrupts should therefore be disabled before entering module stop mode.

# Section 10 8-Bit Timers

## 10.1 Overview

The H8S/2345 Series includes an 8-bit timer module with two channels (TMR0 and TMR1). Each channel has an 8-bit counter (TCNT) and two time constant registers (TCORA and TCORB) that are constantly compared with the TCNT value to detect compare match events. The 8-bit timer module can thus be used for a variety of functions, including pulse output with an arbitrary duty cycle.

### 10.1.1 Features

The features of the 8-bit timer module are listed below.

- Selection of four clock sources
  The counters can be driven by one of three internal clock signals (ø/8, ø/64, or ø/8192) or an external clock input (enabling use as an external event counter).
- Selection of three ways to clear the counters
  The counters can be cleared on compare match A or B, or by an external reset signal.
- Timer output control by a combination of two compare match signals
  The timer output signal in each channel is controlled by a combination of two independent compare match signals, enabling the timer to generate output waveforms with an arbitrary duty cycle or PWM output.
- Provision for cascading of two channels
  — Operation as a 16-bit timer is possible, using channel 0 for the upper 8 bits and channel 1 for the lower 8 bits (16-bit count mode).
  — Channel 1 can be used to count channel 0 compare matches (compare match count mode).
- Three independent interrupts
  Compare match A and B and overflow interrupts can be requested independently.
- A/D converter conversion start trigger can be generated
  Channel 0 compare match A signal can be used as an A/D converter conversion start trigger.
- Module stop mode can be set
  — As the initial setting, 8-bit timer operation is halted. Register access is enabled by exiting module stop mode.

### 10.1.2 Block Diagram

Figure 10.1 shows a block diagram of the 8-bit timer module.

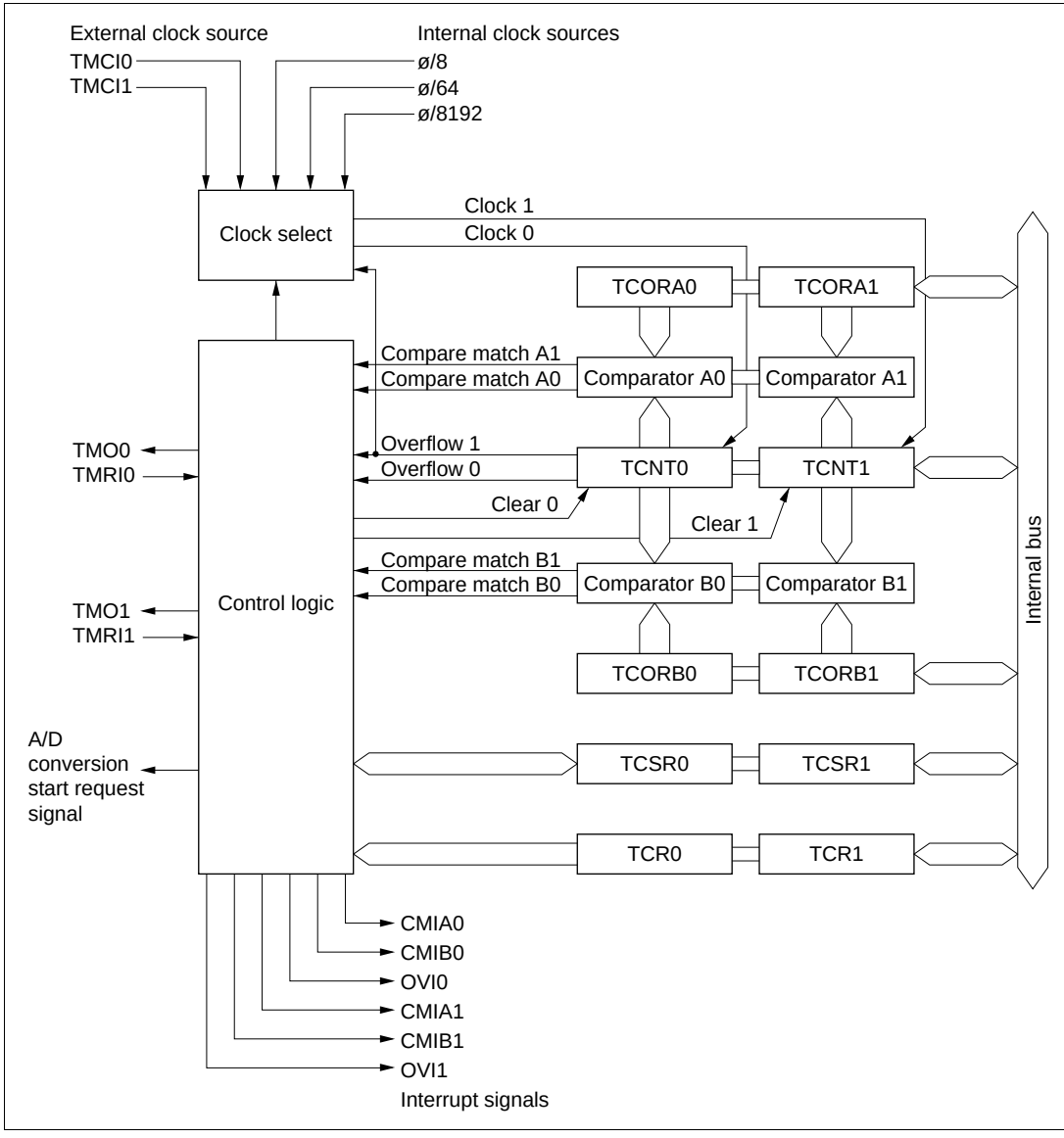



**Figure 10.1 Block Diagram of 8-Bit Timer**

### 10.1.3 Pin Configuration

Table 10.1 summarizes the input and output pins of the 8-bit timer.

**Table 10.1 Input and Output Pins of 8-Bit Timer**

| Channel | Name | Symbol | I/O | Function |
|---|---|---|---|---|
| 0 | Timer output pin 0 | TMO0 | Output | Outputs at compare match |
| | Timer clock input pin 0 | TMCI0 | Input | Inputs external clock for counter |
| | Timer reset input pin 0 | TMRI0 | Input | Inputs external reset to counter |
| 1 | Timer output pin 1 | TMO1 | Output | Outputs at compare match |
| | Timer clock input pin 1 | TMCI1 | Input | Inputs external clock for counter |
| | Timer reset input pin 1 | TMRI1 | Input | Inputs external reset to counter |

### 10.1.4 Register Configuration

Table 10.2 summarizes the registers of the 8-bit timer module.

**Table 10.2 8-Bit Timer Registers**

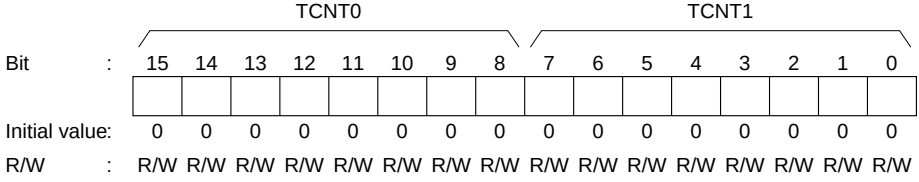
| Channel | Name | Abbreviation | R/W | Initial value | Address*[1] |
|---|---|---|---|---|---|
| 0 | Timer control register 0 | TCR0 | R/W | H'00 | H'FFB0 |
| | Timer control/status register 0 | TCSR0 | R/(W)*[2] | H'00 | H'FFB2 |
| | Time constant register A0 | TCORA0 | R/W | H'FF | H'FFB4 |
| | Time constant register B0 | TCORB0 | R/W | H'FF | H'FFB6 |
| | Timer counter 0 | TCNT0 | R/W | H'00 | H'FFB8 |
| 1 | Timer control register 1 | TCR1 | R/W | H'00 | H'FFB1 |
| | Timer control/status register 1 | TCSR1 | R/(W)*[2] | H'10 | H'FFB3 |
| | Time constant register A1 | TCORA1 | R/W | H'FF | H'FFB5 |
| | Time constant register B1 | TCORB1 | R/W | H'FF | H'FFB7 |
| | Timer counter 1 | TCNT1 | R/W | H'00 | H'FFB9 |
| All | Module stop control register | MSTPCR | R/W | H'3FFF | H'FF3C |

Notes: 1. Lower 16 bits of the address
2. Only 0 can be written to bits 7 to 5, to clear these flags.

Each pair of registers for channel 0 and channel 1 is a 16-bit register with the upper 8 bits for channel 0 and the lower 8 bits for channel 1, so they can be accessed together by word transfer instruction.

## 10.2 Register Descriptions

### 10.2.1 Timer Counters 0 and 1 (TCNT0, TCNT1)




TCNT0 and TCNT1 are 8-bit readable/writable up-counters that increment on pulses generated from an internal or external clock source. This clock source is selected by clock select bits CKS2 to CKS0 of TCR. The CPU can read or write to TCNT0 and TCNT1 at all times.

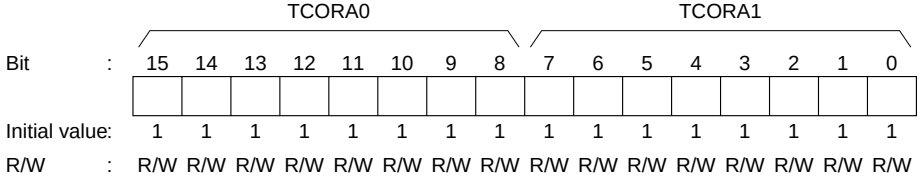
TCNT0 and TCNT1 comprise a single 16-bit register, so they can be accessed together by word transfer instruction.

TCNT0 and TCNT1 can be cleared by an external reset input or by a compare match signal. Which signal is to be used for clearing is selected by clock clear bits CCLR1 and CCLR0 of TCR.

When a timer counter overflows from H'FF to H'00, OVF in TCSR is set to 1.

TCNT0 and TCNT1 are each initialized to H'00 by a reset and in hardware standby mode.

### 10.2.2 Time Constant Registers A0 and A1 (TCORA0, TCORA1)




TCORA0 and TCORA1 are 8-bit readable/writable registers. TCORA0 and TCORA1 comprise a single 16-bit register so they can be accessed together by word transfer instruction.

TCORA is continually compared with the value in TCNT. When a match is detected, the corresponding CMFA flag of TCSR is set. Note, however, that comparison is disabled during the T2 state of a TCOR write cycle.

The timer output can be freely controlled by these compare match signals and the settings of bits OS1 and OS0 of TCSR.

TCORA0 and TCORA1 are each initialized to H'FF by a reset and in hardware standby mode.

### 10.2.3 Time Constant Registers B0 and B1 (TCORB0, TCORB1)

| | | TCORB0 | | | | | | | | TCORB1 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | | |
| Initial value: | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TCORB0 and TCORB1 are 8-bit readable/writable registers. TCORB0 and TCORB1 comprise a single 16-bit register so they can be accessed together by word transfer instruction.

TCORB is continually compared with the value in TCNT. When a match is detected, the corresponding CMFB flag of TCSR is set. Note, however, that comparison is disabled during the T2 state of a TCOR write cycle.

The timer output can be freely controlled by these compare match signals and the settings of output select bits OS3 and OS2 of TCSR.

TCORB0 and TCORB1 are each initialized to H'FF by a reset and in hardware standby mode.

### 10.2.4 Time Control Registers 0 and 1 (TCR0, TCR1)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | CMIEB | CMIEA | OVIE | CCLR1 | CCLR0 | CKS2 | CKS1 | CKS0 |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TCR0 and TCR1 are 8-bit readable/writable registers that select the clock source and the time at which TCNT is cleared, and enable interrupts.

TCR0 and TCR1 are each initialized to H'00 by a reset and in hardware standby mode.

For details of this timing, see section 10.3, Operation.

**Bit 7—Compare Match Interrupt Enable B (CMIEB):** Selects whether CMFB interrupt requests (CMIB) are enabled or disabled when the CMFB flag of TCSR is set to 1.

**Bit 7**

| CMIEB | Description | |
|---|---|---|
| 0 | CMFB interrupt requests (CMIB) are disabled | (Initial value) |
| 1 | CMFB interrupt requests (CMIB) are enabled | |

**Bit 6—Compare Match Interrupt Enable A (CMIEA):** Selects whether CMFA interrupt requests (CMIA) are enabled or disabled when the CMFA flag of TCSR is set to 1.

| Bit 6 | | |
|---|---|---|
| **CMIEA** | **Description** | |
| 0 | CMFA interrupt requests (CMIA) are disabled | (Initial value) |
| 1 | CMFA interrupt requests (CMIA) are enabled | |

**Bit 5—Timer Overflow Interrupt Enable (OVIE):** Selects whether OVF interrupt requests (OVI) are enabled or disabled when the OVF flag of TCSR is set to 1.

| Bit 5 | | |
|---|---|---|
| **OVIE** | **Description** | |
| 0 | OVF interrupt requests (OVI) are disabled | (Initial value) |
| 1 | OVF interrupt requests (OVI) are enabled | |

**Bits 4 and 3—Counter Clear 1 and 0 (CCLR1 and CCLR0):** These bits select the method by which TCNT is cleared: by compare match A or B, or by an external reset input.

| Bit 4 | Bit 3 | | |
|---|---|---|---|
| **CCLR1** | **CCLR0** | **Description** | |
| 0 | 0 | Clear is disabled | (Initial value) |
| | 1 | Clear by compare match A | |
| 1 | 0 | Clear by compare match B | |
| | 1 | Clear by rising edge of external reset input | |

**Bits 2 to 0—Clock Select 2 to 0 (CKS2 to CKS0):** These bits select whether the clock input to TCNT is an internal or external clock.

Three internal clocks can be selected, all divided from the system clock (ø): ø/8, ø/64, and ø/8192. The falling edge of the selected internal clock triggers the count.

When use of an external clock is selected, three types of count can be selected: at the rising edge, the falling edge, and both rising and falling edges.

Some functions differ between channel 0 and channel 1.

| Bit 2 | Bit 1 | Bit 0 | | |
|---|---|---|---|---|
| **CKS2** | **CKS1** | **CKS0** | **Description** | |
| 0 | 0 | 0 | Clock input disabled | (Initial value) |
| | | 1 | Internal clock, counted at falling edge of ø/8 | |
| | 1 | 0 | Internal clock, counted at falling edge of ø/64 | |
| | | 1 | Internal clock, counted at falling edge of ø/8192 | |
| 1 | 0 | 0 | For channel 0: count at TCNT1 overflow signal* | |
| | | | For channel 1: count at TCNT0 compare match A* | |
| | | 1 | External clock, counted at rising edge | |
| | 1 | 0 | External clock, counted at falling edge | |
| | | 1 | External clock, counted at both rising and falling edges | |

Note: * If the count input of channel 0 is the TCNT1 overflow signal and that of channel 1 is the TCNT0 compare match signal, no incrementing clock is generated. Do not use this setting.

### 10.2.5 Timer Control/Status Registers 0 and 1 (TCSR0, TCSR1)

**TCSR0**

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CMFB | CMFA | OVF | ADTE | OS3 | OS2 | OS1 | OS0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/(W)* | R/(W)* | R/(W)* | R/W | R/W | R/W | R/W | R/W |

**TCSR1**

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CMFB | CMFA | OVF | — | OS3 | OS2 | OS1 | OS0 |
| Initial value : | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| R/W : | R/(W)* | R/(W)* | R/(W)* | — | R/W | R/W | R/W | R/W |

Note: * Only 0 can be written to bits 7 to 5, to clear these flags.

TCSR0 and TCSR1 are 8-bit registers that display compare match and overflow statuses, and control compare match output.

TCSR0 is initialized to H'00, and TCSR1 to H'10, by a reset and in hardware standby mode.

**Bit 7—Compare Match Flag B (CMFB):** Status flag indicating whether the values of TCNT and TCORB match.

**Bit 7**

| CMFB | Description |
|---|---|
| 0 | [Clearing conditions] (Initial value)<br>• Cleared by reading CMFB when CMFB = 1, then writing 0 to CMFB<br>• When DTC is activated by CMIB interrupt while DISEL bit of MRB in DTC is 0 |
| 1 | [Setting condition]<br>Set when TCNT matches TCORB |

**Bit 6—Compare Match Flag A (CMFA):** Status flag indicating whether the values of TCNT and TCORA match.

**Bit 6**

| CMFA | Description |
|---|---|
| 0 | [Clearing conditions] (Initial value)<br>• Cleared by reading CMFA when CMFA = 1, then writing 0 to CMFA<br>• When DTC is activated by CMIA interrupt while DISEL bit of MRB in DTC is 0 |
| 1 | [Setting condition]<br>Set when TCNT matches TCORA |

**Bit 5—Timer Overflow Flag (OVF):** Status flag indicating that TCNT has overflowed (changed from H'FF to H'00).

**Bit 5**

| OVF | Description |
|---|---|
| 0 | [Clearing condition] (Initial value)<br>• Cleared by reading OVF when OVF = 1, then writing 0 to OVF |
| 1 | [Setting condition]<br>Set when TCNT overflows from H'FF to H'00 |

**Bit 4—A/D Trigger Enable (ADTE) (TCSR0 Only):** Selects enabling or disabling of A/D converter start requests by compare-match A.

In TCSR1, this bit is reserved: it is always read as 1 and cannot be modified.

| Bit 4 | | |
|---|---|---|
| **ADTE** | **Description** | |
| 0 | A/D converter start requests by compare match A are disabled | (Initial value) |
| 1 | A/D converter start requests by compare match A are enabled | |

**Bits 3 to 0—Output Select 3 to 0 (OS3 to OS0):** These bits specify how the timer output level is to be changed by a compare match of TCOR and TCNT.

Bits OS3 and OS2 select the effect of compare match B on the output level, bits OS1 and OS0 select the effect of compare match A on the output level, and both of them can be controlled independently.

Note, however, that priorities are set such that: toggle output > 1 output > 0 output. If compare matches occur simultaneously, the output changes according to the compare match with the higher priority.

Timer output is disabled when bits OS3 to OS0 are all 0.

After a reset, the timer output is 0 until the first compare match event occurs.

| Bit 3 | Bit 2 | | |
|---|---|---|---|
| **OS3** | **OS2** | **Description** | |
| 0 | 0 | No change when compare match B occurs | (Initial value) |
| | 1 | 0 is output when compare match B occurs | |
| 1 | 0 | 1 is output when compare match B occurs | |
| | 1 | Output is inverted when compare match B occurs (toggle output) | |

| Bit 1 | Bit 0 | | |
|---|---|---|---|
| **OS1** | **OS0** | **Description** | |
| 0 | 0 | No change when compare match A occurs | (Initial value) |
| | 1 | 0 is output when compare match A occurs | |
| 1 | 0 | 1 is output when compare match A occurs | |
| | 1 | Output is inverted when compare match A occurs (toggle output) | |

### 10.2.6 Module Stop Control Register (MSTPCR)

| | | MSTPCRH | | | | | | | | MSTPCRL | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | | |
| Initial value | : | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the MSTP12 bit in MSTPCR is set to 1, the 8-bit timer operation stops at the end of the bus cycle and a transition is made to module stop mode. Registers cannot be read or written to in module stop mode. For details, see section 19.5, Module Stop Mode.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 12—Module Stop (MSTP12):** Specifies the 8-bit timer stop mode.

| Bit 12 | |
|---|---|
| **MSTP12** | **Description** |
| 0 | 8-bit timer module stop mode cleared |
| 1 | 8-bit timer module stop mode set (Initial value) |

## 10.3 Operation

### 10.3.1 TCNT Incrementation Timing

TCNT is incremented by input clock pulses (either internal or external).

**Internal Clock:** Three different internal clock signals (ø/8, ø/64, or ø/8192) divided from the system clock (ø) can be selected, by setting bits CKS2 to CKS0 in TCR. Figure 10.2 shows the count timing.

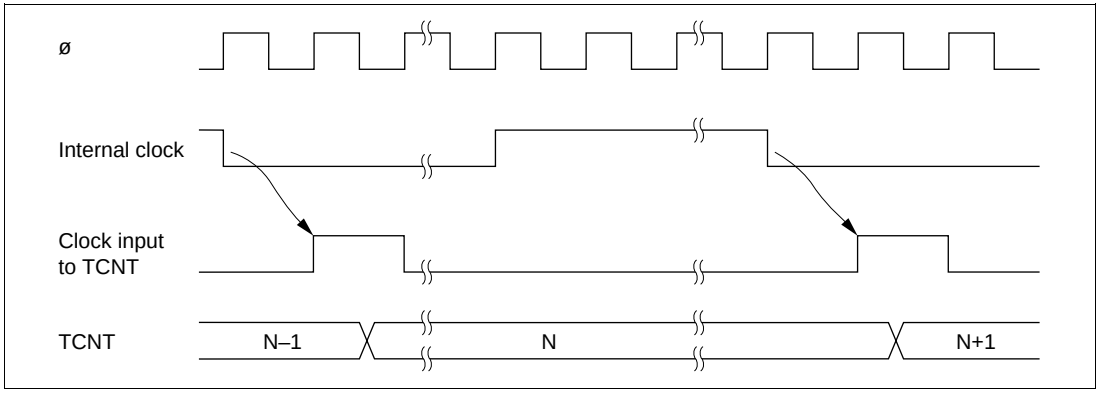



**Figure 10.2 Count Timing for Internal Clock Input**

**External Clock:** Three incrementation methods can be selected by setting bits CKS2 to CKS0 in TCR: at the rising edge, the falling edge, and both rising and falling edges.

Note that the external clock pulse width must be at least 1.5 states for incrementation at a single edge, and at least 2.5 states for incrementation at both edges. The counter will not increment correctly if the pulse width is less than these values.

Figure 10.3 shows the timing of incrementation at both edges of an external clock signal.

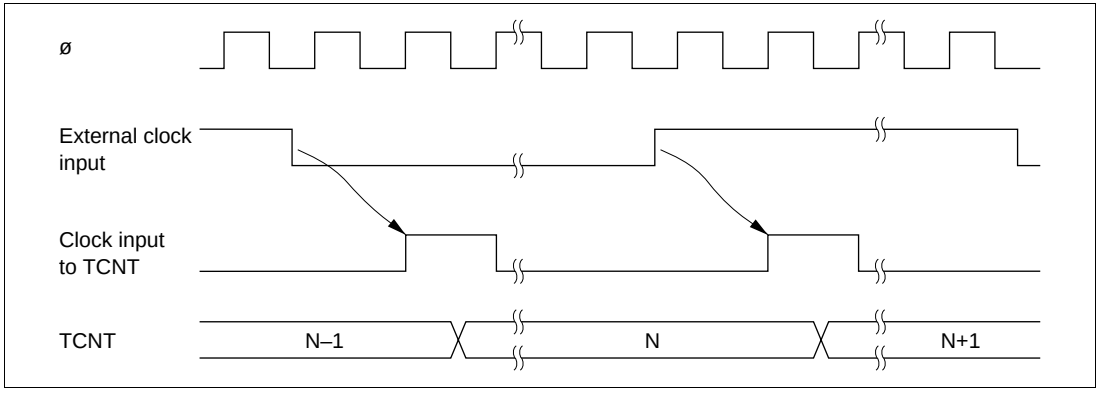



**Figure 10.3 Count Timing for External Clock Input**

### 10.3.2 Compare Match Timing

**Setting of Compare Match Flags A and B (CMFA, CMFB):** The CMFA and CMFB flags in TCSR are set to 1 by a compare match signal generated when the TCOR and TCNT values match. The compare match signal is generated at the last state in which the match is true, just before the timer counter is updated.

Therefore, when TCOR and TCNT match, the compare match signal is not generated until the next incrementation clock input. Figure 10.4 shows this timing.

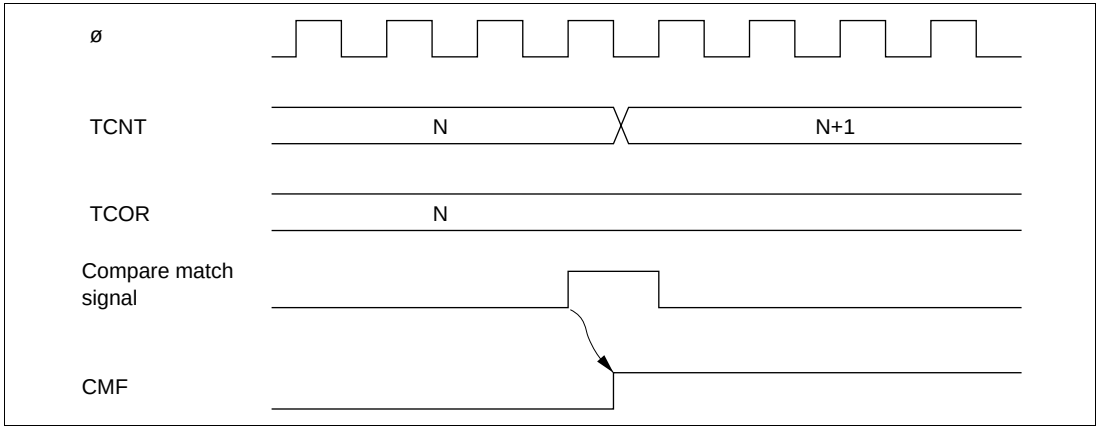



**Figure 10.4 Timing of CMF Setting**

**Timer Output Timing:** When compare match A or B occurs, the timer output changes a specified by bits OS3 to OS0 in TCSR. Depending on these bits, the output can remain the same, change to 0, change to 1, or toggle.

Figure 10.5 shows the timing when the output is set to toggle at compare match A.

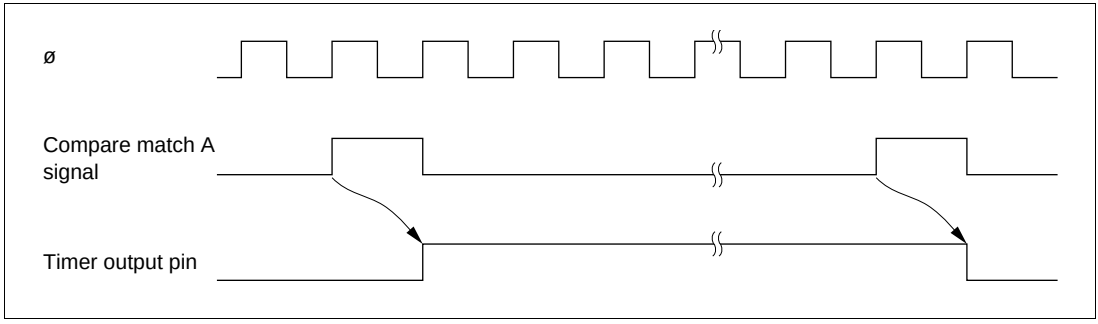



**Figure 10.5 Timing of Timer Output**

**Timing of Compare Match Clear:** The timer counter is cleared when compare match A or B occurs, depending on the setting of the CCLR1 and CCLR0 bits in TCR. Figure 10.6 shows the timing of this operation.

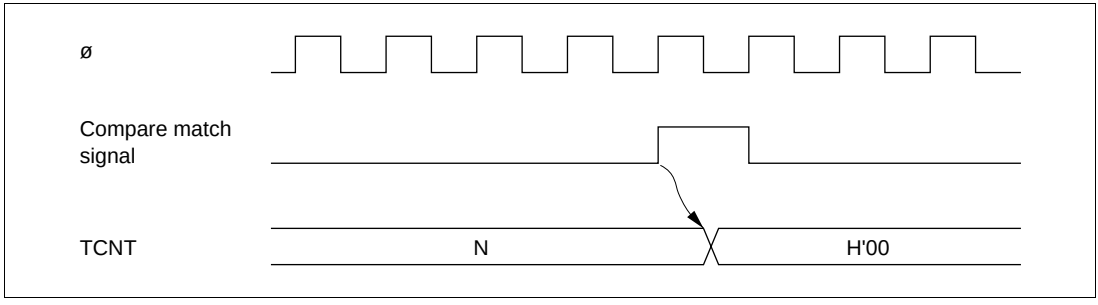



**Figure 10.6 Timing of Compare Match Clear**

### 10.3.3 Timing of External RESET on TCNT

TCNT is cleared at the rising edge of an external reset input, depending on the settings of the CCLR1 and CCLR0 bits in TCR. The clear pulse width must be at least 1.5 states. Figure 10.7 shows the timing of this operation.

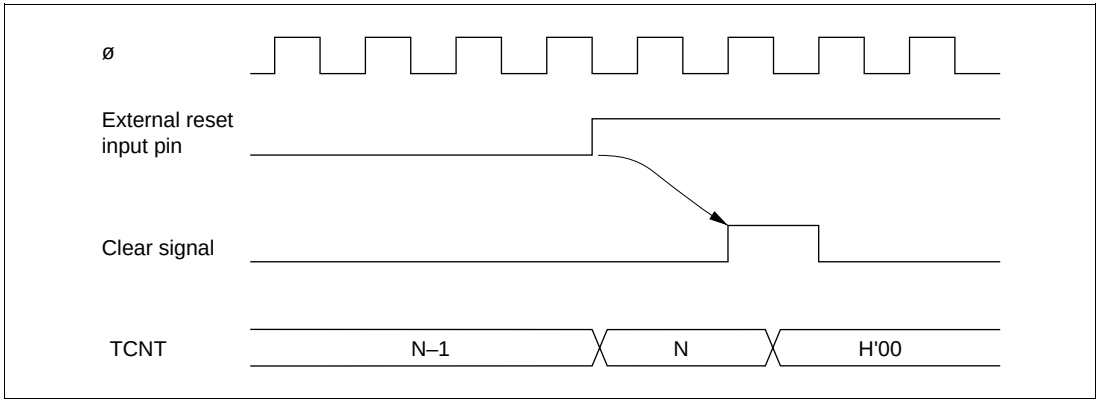



**Figure 10.7 Timing of External Reset**

### 10.3.4 Timing of Overflow Flag (OVF) Setting

The OVF in TCSR is set to 1 when the timer count overflows (changes from H'FF to H'00). Figure 10.8 shows the timing of this operation.

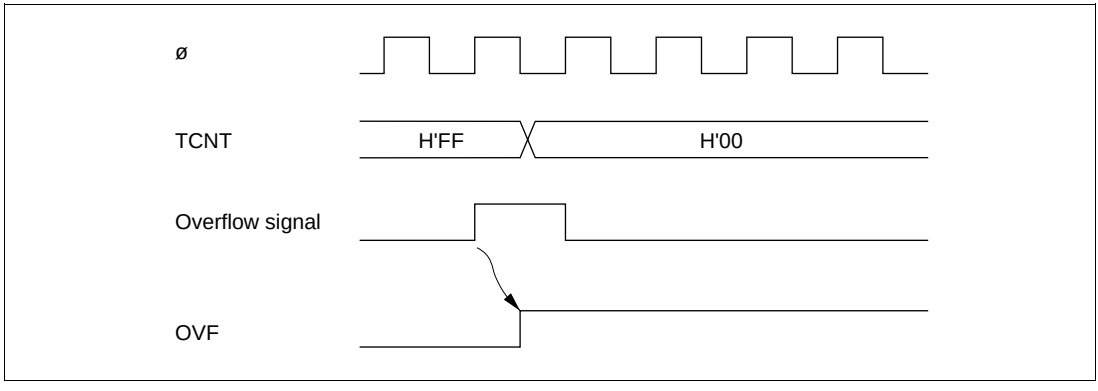



**Figure 10.8 Timing of OVF Setting**

### 10.3.5 Operation with Cascaded Connection

If bits CKS2 to CKS0 in either TCR0 or TCR1 are set to B’100, the 8-bit timers of the two channels are cascaded. With this configuration, a single 16-bit timer could be used (16-bit timer mode) or compare matches of the 8-bit channel 0 could be counted by the timer of channel 1 (compare match counter mode). In this case, the timer operates as below.

**16-Bit Counter Mode:** When bits CKS2 to CKS0 in TCR0 are set to B'100, the timer functions as a single 16-bit timer with channel 0 occupying the upper 8 bits and channel 1 occupying the lower 8 bits.

- Setting of compare match flags
  - — The CMF flag in TCSR0 is set to 1 when a 16-bit compare match event occurs.
  - — The CMF flag in TCSR1 is set to 1 when a lower 8-bit compare match event occurs.

- Counter clear specification
  - — If the CCLR1 and CCLR0 bits in TCR0 have been set for counter clear at compare match, the 16-bit counter (TCNT0 and TCNT1 together) is cleared when a 16-bit compare match event occurs. The 16-bit counter (TCNT0 and TCNT1 together) is cleared even if counter clear by the TMRI0 pin has also been set.
  - — The settings of the CCLR1 and CCLR0 bits in TCR1 are ignored. The lower 8 bits cannot be cleared independently.

- Pin output
  - — Control of output from the TMO0 pin by bits OS3 to OS0 in TCSR0 is in accordance with the 16-bit compare match conditions.
  - — Control of output from the TMO1 pin by bits OS3 to OS0 in TCSR1 is in accordance with the lower 8-bit compare match conditions.

**Compare Match Counter Mode:** When bits CKS2 to CKS0 in TCR1 are B'100, TCNT1 counts compare match A’s for channel 0.

Channels 0 and 1 are controlled independently. Conditions such as setting of the CMF flag, generation of interrupts, output from the TMO pin, and counter clear are in accordance with the settings for each channel.

**Note on Usage:** If the 16-bit counter mode and compare match counter mode are set simultaneously, the input clock pulses for TCNT0 and TCNT1 are not generated and thus the counters will stop operating. Software should therefore avoid using both these modes.

## 10.4 Interrupts

### 10.4.1 Interrupt Sources and DTC Activation

There are three 8-bit timer interrupt sources: CMIA, CMIB, and OVI. Their relative priorities are shown in Table 10.3. Each interrupt source is set as enabled or disabled by the corresponding interrupt enable bit in TCR, and independent interrupt requests are sent for each to the interrupt controller. It is also possible to activate the DTC by means of CMIA and CMIB interrupts.

**Table 10.3 8-Bit Timer Interrupt Sources**

| Interrupt Source | Description | DTC Activation | Priority |
|---|---|---|---|
| CMIA0 | Interrupt by CMFA | Possible | High |
| CMIB0 | Interrupt by CMFB | Possible | |
| OVI0 | Interrupt by OVF | Not possible | |
| CMIA1 | Interrupt by CMFA | Possible | |
| CMIB1 | Interrupt by CMFB | Possible | |
| OVI1 | Interrupt by OVF | Not possible | Low |

Note: This table shows the initial state immediately after a reset. The relative channel priorities can be changed by the interrupt controller.

### 10.4.2 A/D Converter Activation

The A/D converter can be activated only by channel 0 compare match A.

If the ADTE bit in TCSR0 is set to 1 when the CMFA flag is set to 1 by the occurrence of channel 0 compare match A, a request to start A/D conversion is sent to the A/D converter. If the 8-bit timer conversion start trigger has been selected on the A/D converter side at this time, A/D conversion is started.

## 10.5 Sample Application

In the example below, the 8-bit timer is used to generate a pulse output with a selected duty cycle, as shown in figure 10.9. The control bits are set as follows:

[1] In TCR, bit CCLR1 is cleared to 0 and bit CCLR0 is set to 1 so that the timer counter is cleared when its value matches the constant in TCORA.

[2] In TCSR, bits OS3 to OS0 are set to B'0110, causing the output to change to 1 at a TCORA compare match and to 0 at a TCORB compare match.

With these settings, the 8-bit timer provides output of pulses at a rate determined by TCORA with a pulse width determined by TCORB. No software intervention is required.

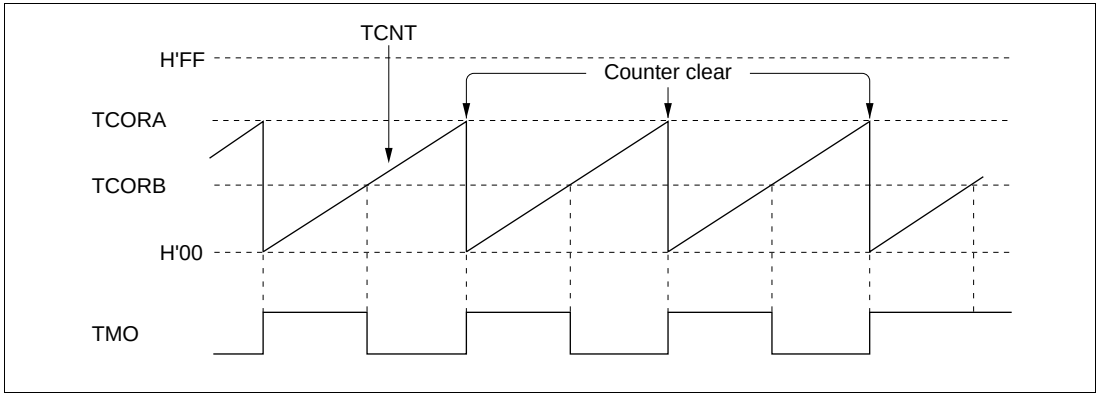



**Figure 10.9 Example of Pulse Output**

## 10.6 Usage Notes

Application programmers should note that the following kinds of contention can occur in the 8-bit timer.

### 10.6.1 Contention between TCNT Write and Clear

If a timer counter clock pulse is generated during the $T_2$ state of a TCNT write cycle, the clear takes priority, so that the counter is cleared and the write is not performed.

Figure 10.10 shows this operation.

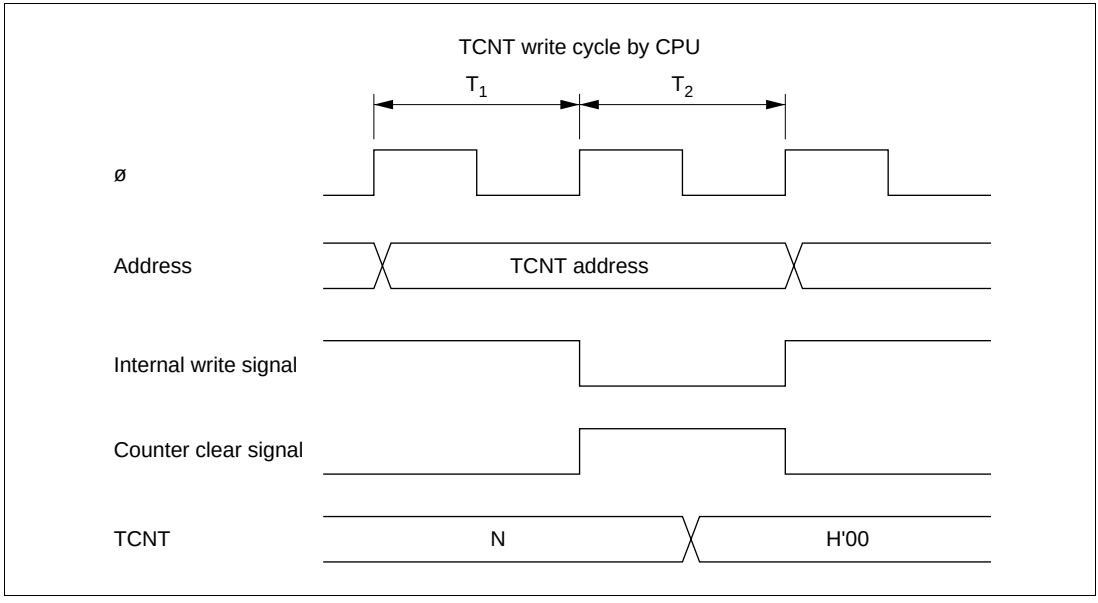



**Figure 10.10 Contention between TCNT Write and Clear**

### 10.6.2 Contention between TCNT Write and Increment

If a timer counter clock pulse is generated during the $T_2$ state of a TCNT write cycle, the write takes priority and the counter is not incremented.

Figure 10.11 shows this operation.

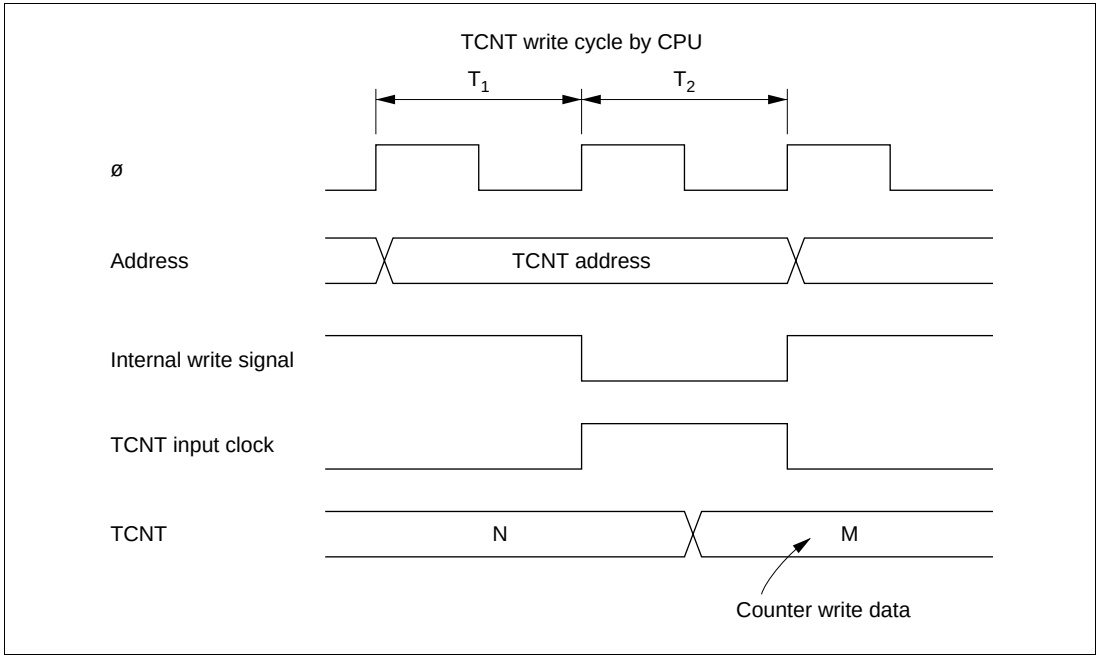



**Figure 10.11 Contention between TCNT Write and Increment**

### 10.6.3 Contention between TCOR Write and Compare Match

During the $T_2$ state of a TCOR write cycle, the TCOR write has priority and the compare match signal is disabled even if a compare match event occurs.

Figure 10.12 shows this operation.

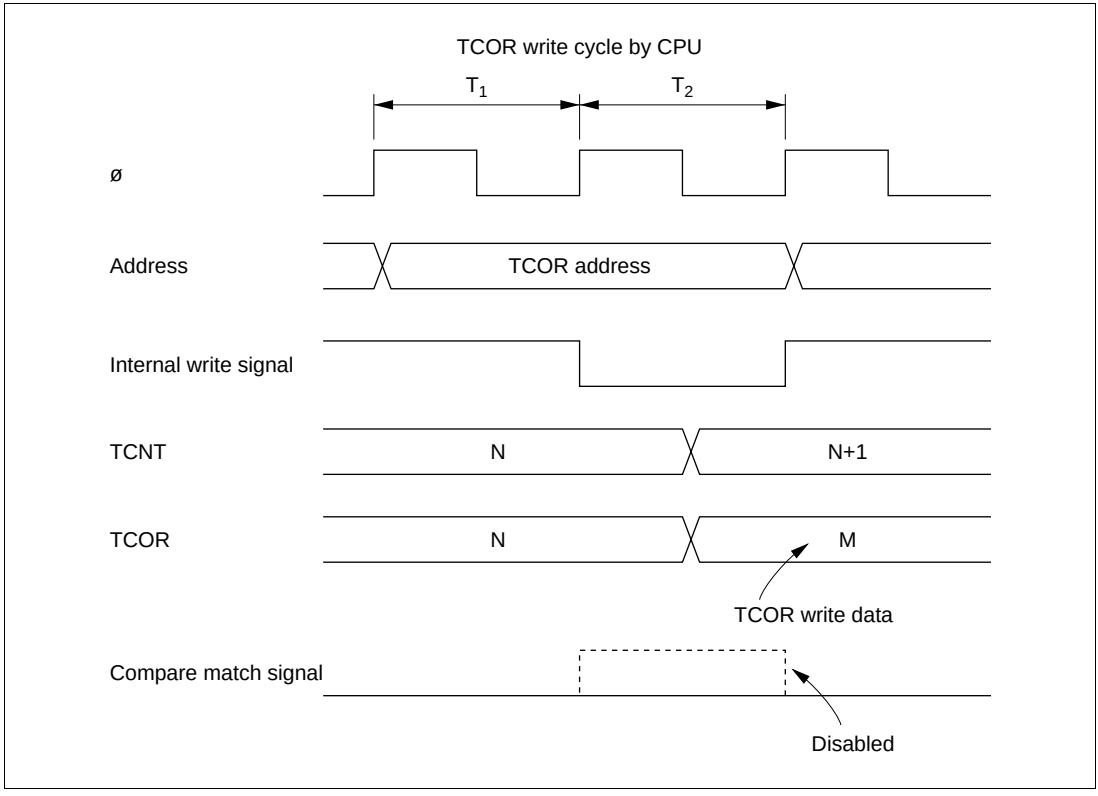



**Figure 10.12 Contention between TCOR Write and Compare Match**

### 10.6.4 Contention between Compare Matches A and B

If compare match events A and B occur at the same time, the 8-bit timer operates in accordance with the priorities for the output statuses set for compare match A and compare match B, as shown in table 10.4.

**Table 10.4 Timer Output Priorities**

| Output Setting | Priority |
|---|---|
| Toggle output | High |
| 1 output | ↑ |
| 0 output | |
| No change | Low |

### 10.6.5 Switching of Internal Clocks and TCNT Operation

TCNT may increment erroneously when the internal clock is switched over. Table 10.5 shows the relationship between the timing at which the internal clock is switched (by writing to the CKS1 and CKS0 bits) and the TCNT operation.

When the TCNT clock is generated from an internal clock, the falling edge of the internal clock pulse is detected. If clock switching causes a change from high to low level, as shown in case 3 in table 10.5, a TCNT clock pulse is generated on the assumption that the switchover is a falling edge. This increments TCNT.

The erroneous incrementation can also happen when switching between internal and external clocks.

**Table 10.5 Switching of Internal Clock and TCNT Operation**

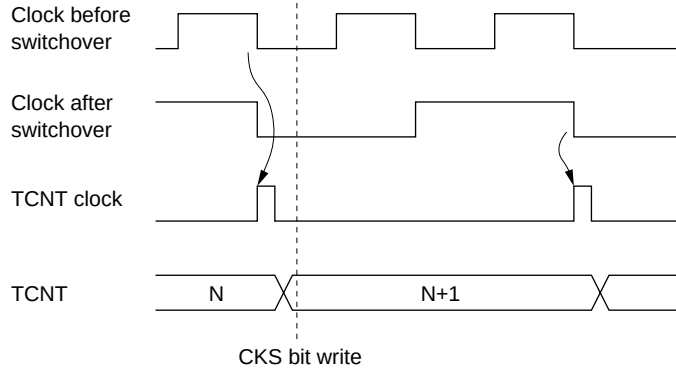
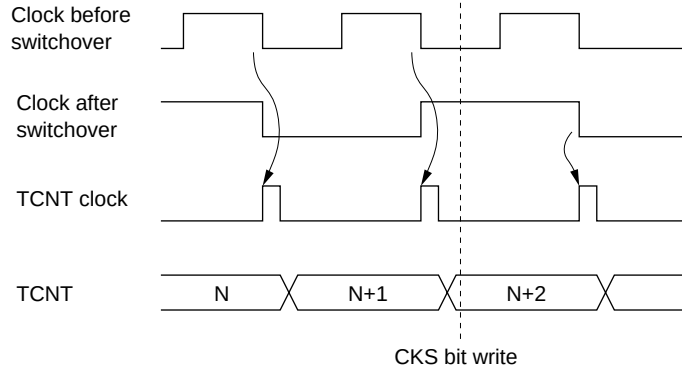
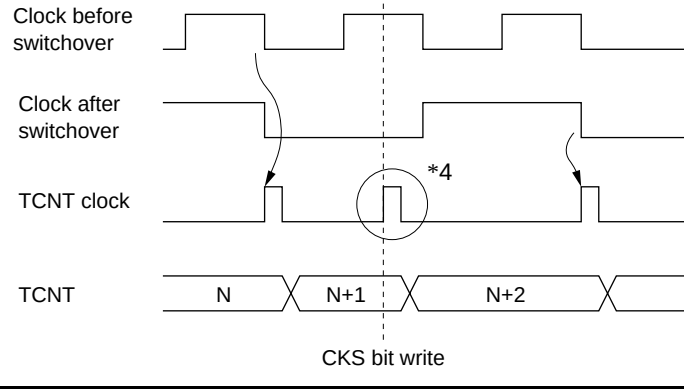
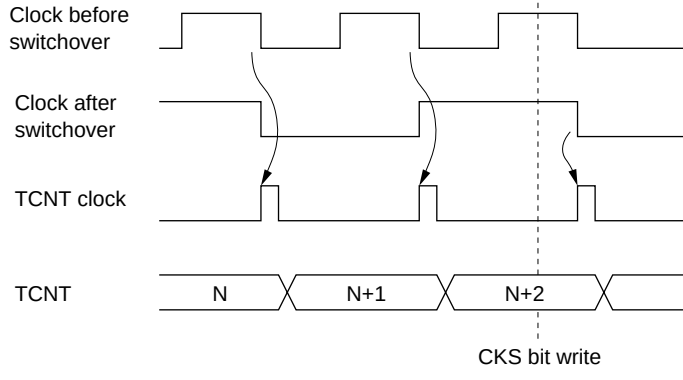
| No. | Timing of Switchover by Means of CKS1 and CKS0 Bits | TCNT Clock Operation |
|---|---|---|
| 1 | Switching from low to low*[1] |  |
| 2 | Switching from low to high*[2] |  |
| 3 | Switching from high to low*[3] |  |

**Table 10.5 Switching of Internal Clock and TCNT Operation (cont)**

| No. | Timing of Switchover by Means of CKS1 and CKS0 Bits | TCNT Clock Operation |
|---|---|---|
| 4 | Switching from high to high |  |



Notes: 1. Includes switching from low to stop, and from stop to low.
2. Includes switching from stop to high.
3. Includes switching from high to stop.
4. Generated on the assumption that the switchover is a falling edge; TCNT is incremented.

### 10.6.6 Usage Note

**Interrupts and Module Stop Mode:** If module stop mode is entered when an interrupt has been requested, it will not be possible to clear the CPU interrupt source or DTC activation source. Interrupts should therefore be disabled before entering module stop mode.

# Section 11 Watchdog Timer

## 11.1 Overview

The H8S/2345 Series has a single-channel on-chip watchdog timer (WDT) for monitoring system operation. The WDT outputs an overflow signal ($\overline{\text{WDTOVF}}$)* if a system crash prevents the CPU from writing to the timer counter, allowing it to overflow. At the same time, the WDT can also generate an internal reset signal for the H8S/2345 Series.

When this watchdog function is not needed, the WDT can be used as an interval timer. In interval timer operation, an interval timer interrupt is generated each time the counter overflows.

### 11.1.1 Features

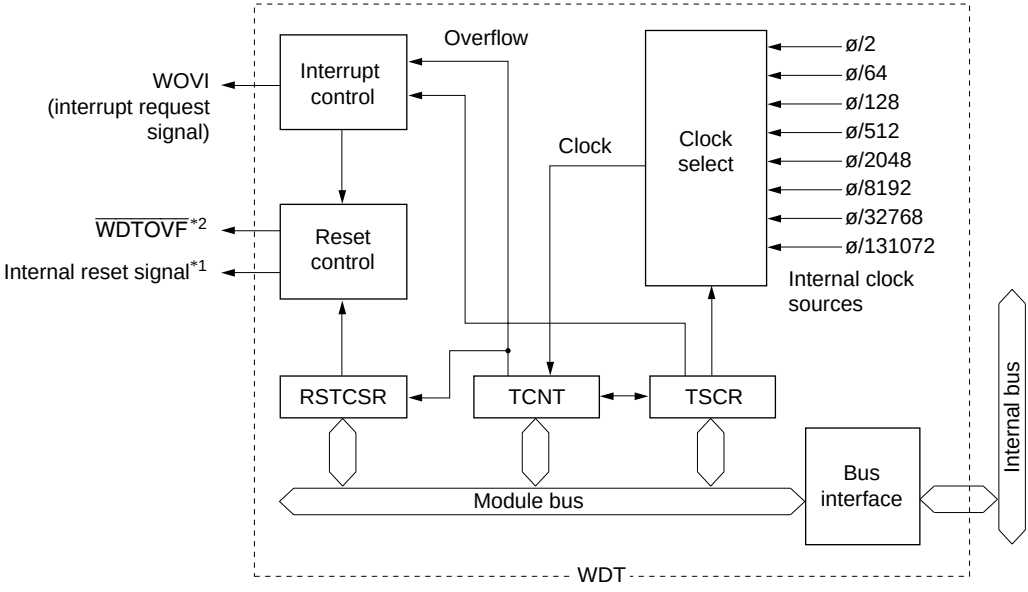
WDT features are listed below.

- Switchable between watchdog timer mode and interval timer mode
- $\overline{\text{WDTOVF}}$ output* when in watchdog timer mode
  If the counter overflows, the WDT outputs $\overline{\text{WDTOVF}}$.* It is possible to select whether or not the entire H8S/2345 Series is reset at the same time. This internal reset can be a power-on reset or a manual reset.
- Interrupt generation when in interval timer mode
  If the counter overflows, the WDT generates an interval timer interrupt.
- Choice of eight counter clock sources.

Note: * The $\overline{\text{WDTOVF}}$ pin function is not supported by the F-ZTAT version.

### 11.1.2 Block Diagram

Figure 11.1 shows a block diagram of the WDT.




**Legend**
TCSR : Timer control/status register
TCNT : Timer counter
RSTCSR : Reset control/status register

Notes: 1. The type of internal reset signal depends on a register setting. Either power-on reset or manual reset can be selected.
2. The $\overline{\text{WDTOVF}}$ pin function is not supported by the F-ZTAT version.

**Figure 11.1 Block Diagram of WDT**

### 11.1.3 Pin Configuration

Table 11.1 describes the WDT output pin.

**Table 11.1 WDT Pin**

| Name | Symbol | I/O | Function |
|---|---|---|---|
| Watchdog timer overflow | $\overline{\text{WDTOVF}}$* | Output | Outputs counter overflow signal in watchdog timer mode |

Note: * The $\overline{\text{WDTOVF}}$ pin function is not supported by the F-ZTAT version.

### 11.1.4 Register Configuration

The WDT has three registers, as summarized in table 11.2. These registers control clock selection, WDT mode switching, and the reset signal.

**Table 11.2 WDT Registers**

| | | | | Address*[1] | |
|---|---|---|---|---|---|
| **Name** | **Abbreviation** | **R/W** | **Initial Value** | **Write*[2]** | **Read** |
| Timer control/status register | TCSR | R/(W)*[3] | H'18 | H'FFBC | H'FFBC |
| Timer counter | TCNT | R/W | H'00 | H'FFBC | H'FFBD |
| Reset control/status register | RSTCSR | R/(W)*[3] | H'1F | H'FFBE | H'FFBF |

Notes: 1. Lower 16 bits of the address.
2. For details of write operations, see section 11.2.4, Notes on Register Access.
3. Only a write of 0 is permitted to bit 7, to clear the flag.

## 11.2 Register Descriptions

### 11.2.1 Timer Counter (TCNT)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TCNT is an 8-bit readable/writable*[1] up-counter.

When the TME bit is set to 1 in TCSR, TCNT starts counting pulses generated from the internal clock source selected by bits CKS2 to CKS0 in TCSR. When the count overflows (changes from H'FF to H'00), either the watchdog timer overflow signal ($\overline{\text{WDTOVF}}$)*[2] or an interval timer interrupt (WOVI) is generated, depending on the mode selected by the WT/$\overline{\text{IT}}$ bit in TCSR.

TCNT is initialized to H'00 by a reset, in hardware standby mode, or when the TME bit is cleared to 0. It is not initialized in software standby mode.

Note: 1. The method for writing to TCNT is different from that for general registers to prevent inadvertent overwriting. For details see section 11.2.4, Notes on Register Access.
2. The $\overline{\text{WDTOVF}}$ pin function is not supported by the F-ZTAT version.

### 11.2.2 Timer Control/Status Register (TCSR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | OVF | WT/$\overline{\text{IT}}$ | TME | — | — | CKS2 | CKS1 | CKS0 |
| Initial value | : | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| R/W | : | R/(W)* | R/W | R/W | — | — | R/W | R/W | R/W |

Note: * Can only be written with 0 for flag clearing.

TCSR is an 8-bit readable/writable* register. Its functions include selecting the clock source to be input to TCNT, and the timer mode.

TCR is initialized to H'18 by a reset and in hardware standby mode. It is not initialized in software standby mode.

Note: * The method for writing to TCSR is different from that for general registers to prevent inadvertent overwriting. For details see section 11.2.4, Notes on Register Access.

**Bit 7—Overflow Flag (OVF):** Indicates that TCNT has overflowed from H'FF to H'00, when in interval timer mode. This flag cannot be set during watchdog timer operation.

**Bit 7**

| OVF | Description | |
|---|---|---|
| 0 | [Clearing condition] | |
| | Cleared by reading TCSR when OVF = 1, then writing 0 to OVF | (Initial value) |
| 1 | [Setting condition] | |
| | Set when TCNT overflows (changes from H'FF to H'00) in interval timer mode | |

**Bit 6—Timer Mode Select (WT/$\overline{\text{IT}}$):** Selects whether the WDT is used as a watchdog timer or interval timer. If used as an interval timer, the WDT generates an interval timer interrupt request (WOVI) when TCNT overflows. If used as a watchdog timer, the WDT generates the $\overline{\text{WDTOVF}}$ signal* when TCNT overflows.

Note: * The $\overline{\text{WDTOVF}}$ pin function is not supported by the F-ZTAT version.

**Bit 6**

| WT/$\overline{\text{IT}}$ | Description | |
|---|---|---|
| 0 | Interval timer: Sends the CPU an interval timer interrupt request (WOVI) when TCNT overflows | (Initial value) |
| 1 | Watchdog timer: Generates the $\overline{\text{WDTOVF}}$ signal*[1] when TCNT overflows*[2] | |

Notes: 1. The $\overline{\text{WDTOVF}}$ pin function is not supported by the F-ZTAT version.
2. For details of the case where TCNT overflows in watchdog timer mode, see section 11.2.3, Reset Control/Status Register (RSTCSR).

**Bit 5—Timer Enable (TME):** Selects whether TCNT runs or is halted.

**Bit 5**

| TME | Description | |
|---|---|---|
| 0 | TCNT is initialized to H'00 and halted | (Initial value) |
| 1 | TCNT counts | |

**Bits 4 and 3—Reserved:** Read-only bits, always read as 1.

**Bits 2 to 0:** Clock Select 2 to 0 (CKS2 to CKS0): These bits select one of eight internal clock sources, obtained by dividing the system clock (ø), for input to TCNT.

| Bit 2 | Bit 1 | Bit 0 | Description | |
|---|---|---|---|---|
| **CKS2** | **CKS1** | **CKS0** | **Clock** | **Overflow Period (when ø = 20 MHz)*** |
| 0 | 0 | 0 | ø/2 (initial value) | 25.6 µs |
| | | 1 | ø/64 | 819.2 µs |
| | 1 | 0 | ø/128 | 1.6 ms |
| | | 1 | ø/512 | 6.6 ms |
| 1 | 0 | 0 | ø/2048 | 26.2 ms |
| | | 1 | ø/8192 | 104.9 ms |
| | 1 | 0 | ø/32768 | 419.4 ms |
| | | 1 | ø/131072 | 1.68 s |

Note: * The overflow period is the time from when TCNT starts counting up from H'00 until overflow occurs.

### 11.2.3 Reset Control/Status Register (RSTCSR)

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WOVF | RSTE | RSTS | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| R/W : | R/(W)* | R/W | R/W | — | — | — | — | — |

Note: * Can only be written with 0 for flag clearing.

RSTCSR is an 8-bit readable/writable* register that controls the generation of the internal reset signal when TCNT overflows, and selects the type of internal reset signal.

RSTCSR is initialized to H'1F by a reset signal from the $\overline{\text{RES}}$ pin, but not by the WDT internal reset signal caused by overflows.

Note: * The method for writing to RSTCSR is different from that for general registers to prevent inadvertent overwriting. For details see section 11.2.4, Notes on Register Access.

**Bit 7—Watchdog Overflow Flag (WOVF):** Indicates that TCNT has overflowed (changed from H'FF to H'00) during watchdog timer operation. This bit is not set in interval timer mode.

| Bit 7 | | |
|---|---|---|
| **WOVF** | **Description** | |
| 0 | [Clearing condition] | (Initial value) |
| | Cleared by reading TCSR when WOVF = 1, then writing 0 to WOVF | |
| 1 | [Setting condition] | |
| | Set when TCNT overflows (changed from H'FF to H'00) during watchdog timer operation | |

**Bit 6—Reset Enable (RSTE):** Specifies whether or not a reset signal is generated in the H8S/2345 Series if TCNT overflows during watchdog timer operation.

| Bit 6 | | |
|---|---|---|
| **RSTE** | **Description** | |
| 0 | Reset signal is not generated if TCNT overflows* | (Initial value) |
| 1 | Reset signal is generated if TCNT overflows | |

Note: * The modules within the H8S/2345 Series are not reset, but TCNT and TCSR within the WDT are reset.

**Bit 5—Reset Select (RSTS):** Selects the type of internal reset generated if TCNT overflows during watchdog timer operation.

For details of the types of resets, see section 4, Exception Handling.

| Bit 5 | | |
|---|---|---|
| **RSTS** | **Description** | |
| 0 | Power-on reset | (Initial value) |
| 1 | Manual reset | |

**Bits 4 to 0—Reserved:** Read-only bits, always read as 1.

### 11.2.4 Notes on Register Access

The watchdog timer's TCNT, TCSR, and RSTCSR registers differ from other registers in being more difficult to write to. The procedures for writing to and reading these registers are given below.

**Writing to TCNT and TCSR:** These registers must be written to by a word transfer instruction. They cannot be written to with byte instructions.

Figure 11.2 shows the format of data written to TCNT and TCSR. TCNT and TCSR both have the same write address. For a write to TCNT, the upper byte of the written word must contain H'5A and the lower byte must contain the write data. For a write to TCSR, the upper byte of the written word must contain H'A5 and the lower byte must contain the write data. This transfers the write data from the lower byte to TCNT or TCSR.

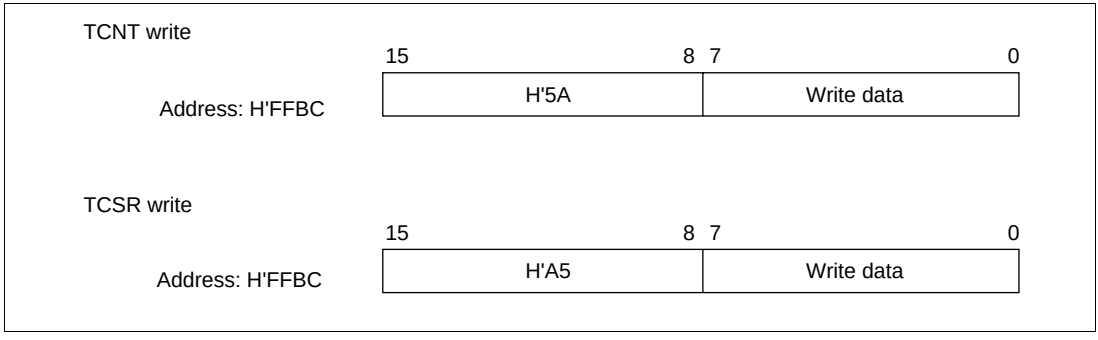



**Figure 11.2 Format of Data Written to TCNT and TCSR**

**Writing to RSTCSR:** RSTCSR must be written to by word transfer instruction to address H'FFBE. It cannot be written to with byte instructions.

Figure 11.3 shows the format of data written to RSTCSR. The method of writing 0 to the WOVF bit differs from that for writing to the RSTE and RSTS bits.

To write 0 to the WOVF bit, the write data must have H'A5 in the upper byte and H'00 in the lower byte. This clears the WOVF bit to 0, but has no effect on the RSTE and RSTS bits. To write to the RSTE and RSTS bits, the upper byte must contain H'5A and the lower byte must contain the write data. This writes the values in bits 6 and 5 of the lower byte into the RSTE and RSTS bits, but has no effect on the WOVF bit.

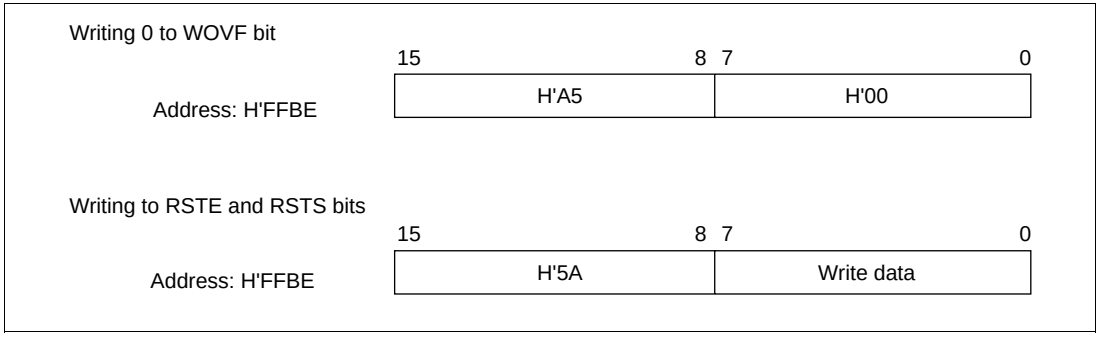



**Figure 11.3 Format of Data Written to RSTCSR**

**Reading TCNT, TCSR, and RSTCSR:** These registers are read in the same way as other registers. The read addresses are H'FFBC for TCSR, H'FFBD for TCNT, and H'FFBF for RSTCSR.

## 11.3 Operation

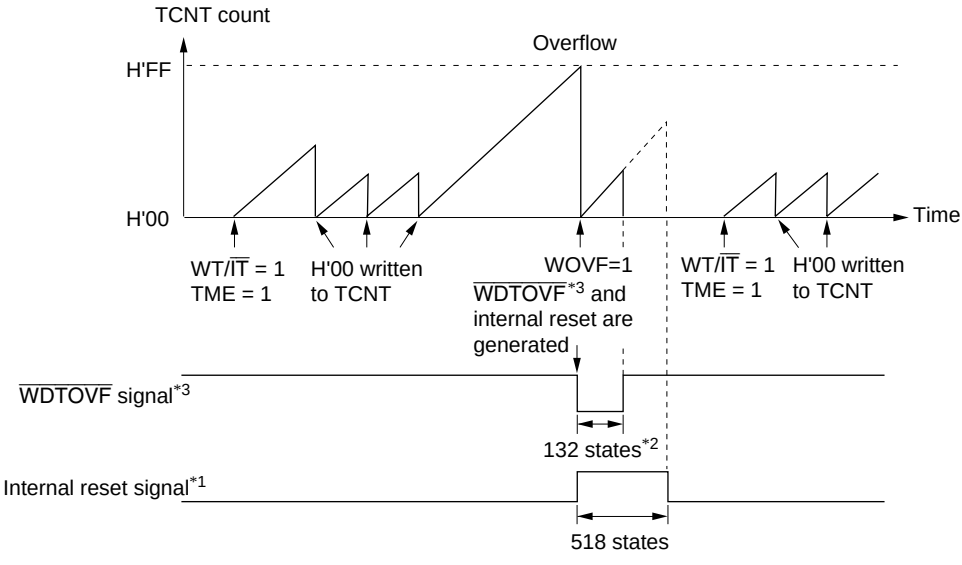
### 11.3.1 Watchdog Timer Operation

To use the WDT as a watchdog timer, set the WT/$\overline{IT}$ and TME bits to 1. Software must prevent TCNT overflows by rewriting the TCNT value (normally be writing H'00) before overflows occurs. This ensures that TCNT does not overflow while the system is operating normally. If TCNT overflows without being rewritten because of a system crash or other error, the $\overline{WDTOVF}$ signal* is output. This is shown in figure 11.4. This $\overline{WDTOVF}$ signal* can be used to reset the system. The $\overline{WDTOVF}$ signal* is output for 132 states when RSTE = 1, and for 130 states when RSTE = 0.

If TCNT overflows when 1 is set in the RSTE bit in RSTCSR, a signal that resets the H8S/2345 Series internally is generated at the same time as the $\overline{WDTOVF}$ signal*. This reset can be selected as a power-on reset or a manual reset, depending on the setting of the RSTS bit in RSTCSR. The internal reset signal is output for 518 states.

If a reset caused by a signal input to the $\overline{RES}$ pin occurs at the same time as a reset caused by a WDT overflow, the $\overline{RES}$ pin reset has priority and the WOVF bit in RSTCSR is cleared to 0.

Note: * The $\overline{WDTOVF}$ pin function is not supported by the F-ZTAT version.




**Legend**

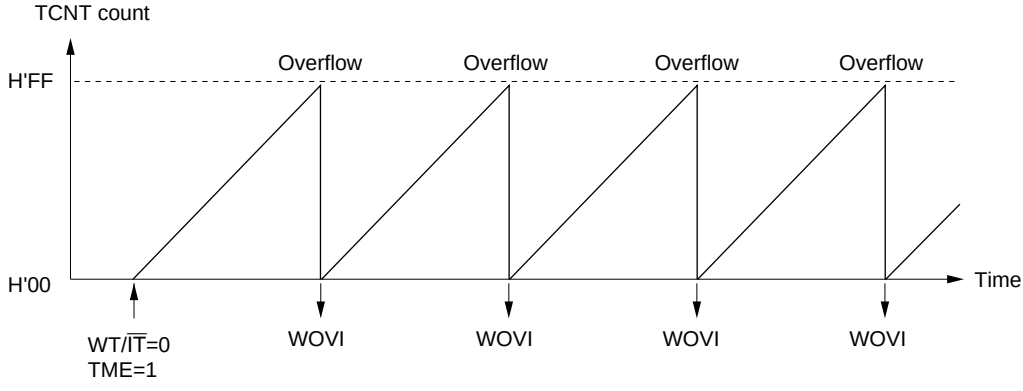
WT/$\overline{\text{IT}}$ : Timer mode select bit
TME : Timer enable bit

Notes: 1. The internal reset signal is generated only if the RSTE bit is set to 1.
2. 130 states when the RSTE bit is cleared to 0.
3. The $\overline{\text{WDTOVF}}$ pin function is not supported by the F-ZTAT version.

**Figure 11.4 Watchdog Timer Operation**

### 11.3.2 Interval Timer Operation

To use the WDT as an interval timer, clear the WT/$\overline{IT}$ bit in TCSR to 0 and set the TME bit to 1. An interval timer interrupt (WOVI) is generated each time TCNT overflows, provided that the WDT is operating as an interval timer, as shown in figure 11.5. This function can be used to generate interrupt requests at regular intervals.




**Legend**
WOVI: Interval timer interrupt request generation

**Figure 11.5 Interval Timer Operation**

### 11.3.3 Timing of Setting Overflow Flag (OVF)

The OVF flag is set to 1 if TCNT overflows during interval timer operation. At the same time, an interval timer interrupt (WOVI) is requested. This timing is shown in figure 11.6.

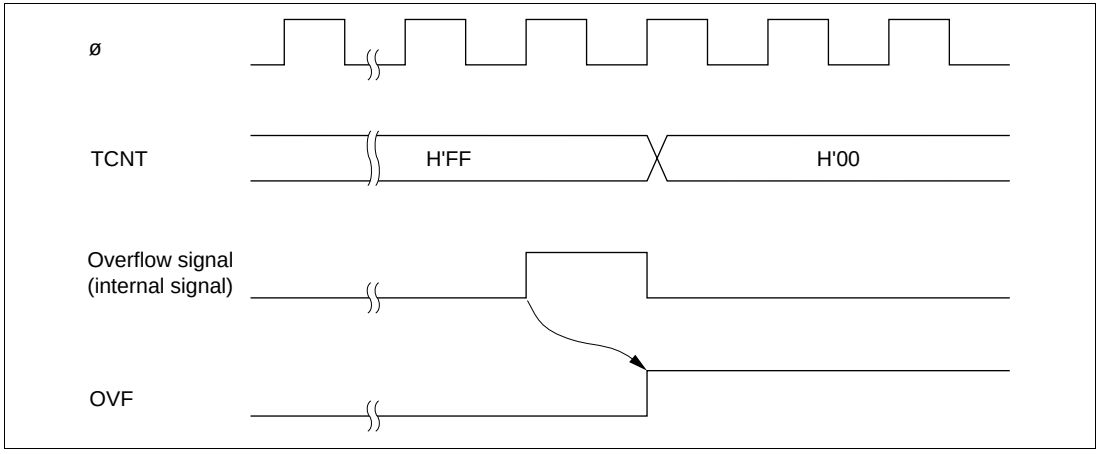



**Figure 11.6 Timing of Setting of OVF**

### 11.3.4 Timing of Setting of Watchdog Timer Overflow Flag (WOVF)

The WOVF flag is set to 1 if TCNT overflows during watchdog timer operation. At the same time, the $\overline{\text{WDTOVF}}$ signal* goes low. If TCNT overflows while the RSTE bit in RSTCSR is set to 1, an internal reset signal is generated for the entire H8S/2345 Series chip. Figure 11.7 shows the timing in this case.

Note: * The $\overline{\text{WDTOVF}}$ pin function is not supported by the F-ZTAT version.

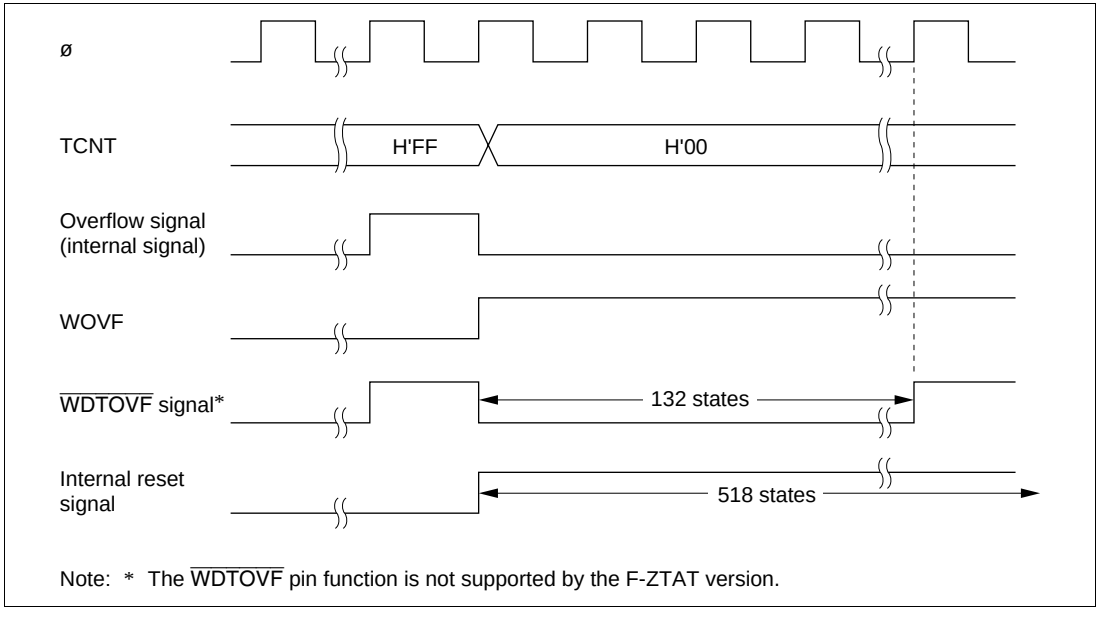



**Figure 11.7 Timing of Setting of WOVF**

## 11.4 Interrupts

During interval timer mode operation, an overflow generates an interval timer interrupt (WOVI). The interval timer interrupt is requested whenever the OVF flag is set to 1 in TCSR.

## 11.5 Usage Notes

### 11.5.1 Contention between Timer Counter (TCNT) Write and Increment

If a timer counter clock pulse is generated during the $T_2$ state of a TCNT write cycle, the write takes priority and the timer counter is not incremented. Figure 11.8 shows this operation.

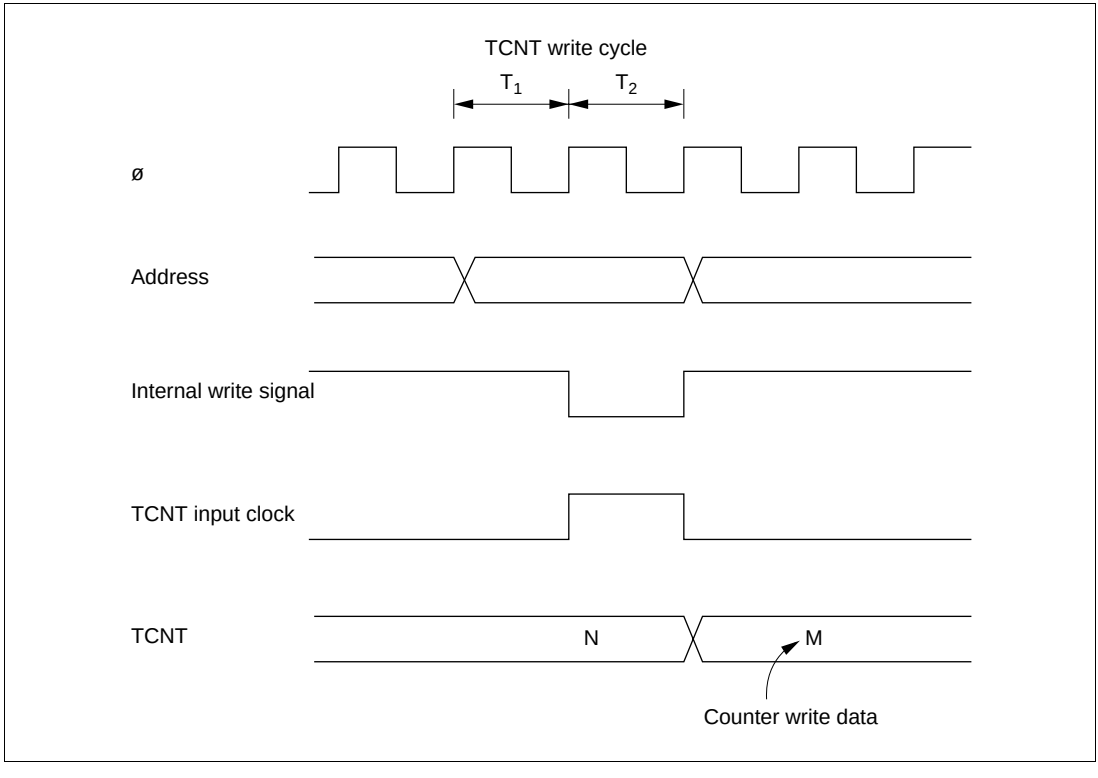



**Figure 11.8 Contention between TCNT Write and Increment**

### 11.5.2 Changing Value of CKS2 to CKS0

If bits CKS2 to CKS0 in TCSR are written to while the WDT is operating, errors could occur in the incrementation. Software must stop the watchdog timer (by clearing the TME bit to 0) before changing the value of bits CKS2 to CKS0.

### 11.5.3 Switching between Watchdog Timer Mode and Interval Timer Mode

If the mode is switched from watchdog timer to interval timer, or vice versa, while the WDT is operating, errors could occur in the incrementation. Software must stop the watchdog timer (by clearing the TME bit to 0) before switching the mode.

### 11.5.4 System Reset by $\overline{\text{WDTOVF}}$ Signal

If the $\overline{\text{WDTOVF}}$ output signal* is input to the $\overline{\text{RES}}$ pin of the H8S/2345 Series, the H8S/2345 Series will not be initialized correctly. Make sure that the $\overline{\text{WDTOVF}}$ signal* is not input logically to the $\overline{\text{RES}}$ pin. To reset the entire system by means of the $\overline{\text{WDTOVF}}$ signal*, use the circuit shown in figure 11.9.

Note: * The $\overline{\text{WDTOVF}}$ pin function is not supported by the F-ZTAT version.

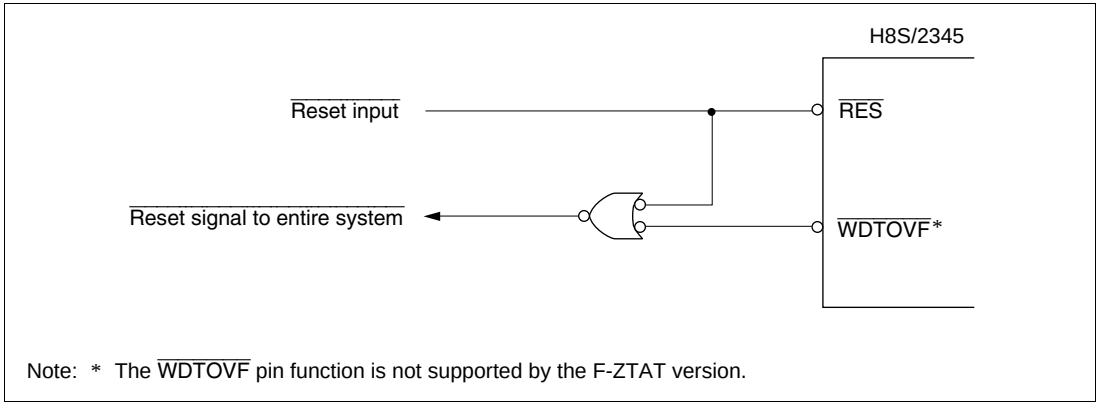



**Figure 11.9 Circuit for System Reset by $\overline{\text{WDTOVF}}$ Signal (Example)**

### 11.5.5 Internal Reset in Watchdog Timer Mode

The H8S/2345 Series is not reset internally if TCNT overflows while the RSTE bit is cleared to 0 during watchdog timer operation, but TCNT and TSCR of the WDT are reset.

TCNT, TCSR, and RSTCR cannot be written to while the $\overline{\text{WDTOVF}}$ signal* is low. Also note that a read of the WOVF flag is not recognized during this period. To clear the WOVF flag, therefore, read TCSR after the $\overline{\text{WDTOVF}}$ signal* goes high, then write 0 to the WOVF flag.

Note: * The $\overline{\text{WDTOVF}}$ pin function is not supported by the F-ZTAT version.

# Section 12 Serial Communication Interface (SCI)

## 12.1 Overview

The H8S/2345 Series is equipped with a 2-channel serial communication interface (SCI). Both channels have the same functions. The SCI can handle both asynchronous and clocked synchronous serial communication. A function is also provided for serial communication between processors (multiprocessor communication function).

### 12.1.1 Features

SCI features are listed below.

- Choice of asynchronous or clocked synchronous serial communication mode

  Asynchronous mode
  - Serial data communication executed using asynchronous system in which synchronization is achieved character by character
  - Serial data communication can be carried out with standard asynchronous communication chips such as a Universal Asynchronous Receiver/Transmitter (UART) or Asynchronous Communication Interface Adapter (ACIA)
  - A multiprocessor communication function is provided that enables serial data communication with a number of processors
  - Choice of 12 serial data transfer formats

    Data length : 7 or 8 bits
    Stop bit length : 1 or 2 bits
    Parity : Even, odd, or none
    Multiprocessor bit : 1 or 0
  - Receive error detection : Parity, overrun, and framing errors
  - Break detection : Break can be detected by reading the RxD pin level directly in case of a framing error

  Clocked Synchronous mode
  - Serial data communication synchronized with a clock

    Serial data communication can be carried out with other chips that have a synchronous communication function
  - One serial data transfer format

    Data length : 8 bits
  - Receive error detection : Overrun errors detected

- Full-duplex communication capability
  - — The transmitter and receiver are mutually independent, enabling transmission and reception to be executed simultaneously
  - — Double-buffering is used in both the transmitter and the receiver, enabling continuous transmission and continuous reception of serial data
- On-chip baud rate generator allows any bit rate to be selected
- Choice of serial clock source: internal clock from baud rate generator or external clock from SCK pin
- Four interrupt sources
  - — Four interrupt sources — transmit-data-empty, transmit-end, receive-data-full, and receive error — that can issue requests independently
  - — The transmit-data-empty interrupt and receive data full interrupts can activate the data transfer controller (DTC) to execute data transfer
- Choice of LSB-first or MSB-first transfer
  - — Can be selected regardless of the communication mode* (except in the case of asynchronous mode bit data)
- Module stop mode can be set
  - — As the initial setting, SCI operation is halted. Register access is enabled by exiting module stop mode.

Note: * Descriptions in this section refer to LSB-first transfer.

### 12.1.2 Block Diagram

Figure 12.1 shows a block diagram of the SCI.

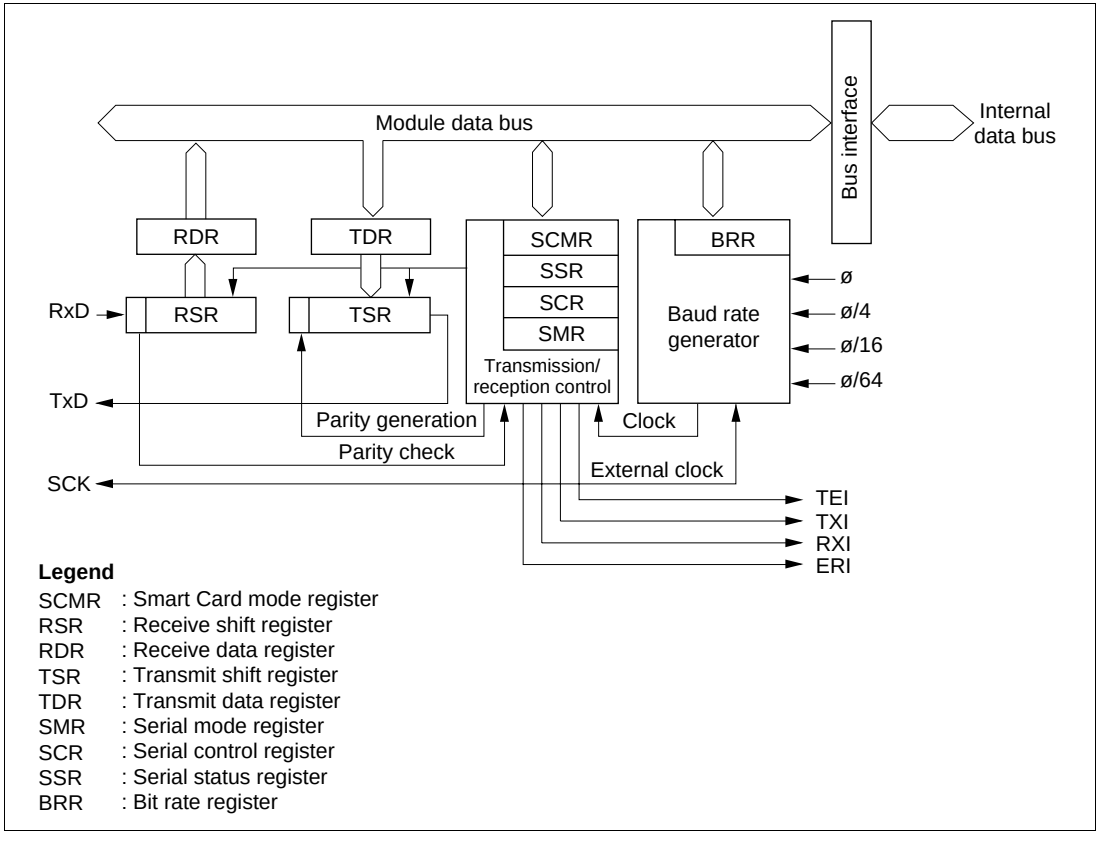



**Figure 12.1 Block Diagram of SCI**

### 12.1.3 Pin Configuration

Table 12.1 shows the serial pins for each SCI channel.

**Table 12.1 SCI Pins**

| Channel | Pin Name | Symbol | I/O | Function |
|---|---|---|---|---|
| 0 | Serial clock pin 0 | SCK0 | I/O | SCI0 clock input/output |
| | Receive data pin 0 | RxD0 | Input | SCI0 receive data input |
| | Transmit data pin 0 | TxD0 | Output | SCI0 transmit data output |
| 1 | Serial clock pin 1 | SCK1 | I/O | SCI1 clock input/output |
| | Receive data pin 1 | RxD1 | Input | SCI1 receive data input |
| | Transmit data pin 1 | TxD1 | Output | SCI1 transmit data output |

### 12.1.4 Register Configuration

The SCI has the internal registers shown in table 12.2. These registers are used to specify asynchronous mode or clocked synchronous mode, the data format , and the bit rate, and to control transmitter/receiver.

**Table 12.2 SCI Registers**

| Channel | Name | Abbreviation | R/W | Initial Value | Address*[1] |
|---|---|---|---|---|---|
| 0 | Serial mode register 0 | SMR0 | R/W | H'00 | H'FF78 |
| | Bit rate register 0 | BRR0 | R/W | H'FF | H'FF79 |
| | Serial control register 0 | SCR0 | R/W | H'00 | H'FF7A |
| | Transmit data register 0 | TDR0 | R/W | H'FF | H'FF7B |
| | Serial status register 0 | SSR0 | R/(W)*[2] | H'84 | H'FF7C |
| | Receive data register 0 | RDR0 | R | H'00 | H'FF7D |
| | Smart card mode register 0 | SCMR0 | R/W | H'F2 | H'FF7E |
| 1 | Serial mode register 1 | SMR1 | R/W | H'00 | H'FF80 |
| | Bit rate register 1 | BRR1 | R/W | H'FF | H'FF81 |
| | Serial control register 1 | SCR1 | R/W | H'00 | H'FF82 |
| | Transmit data register 1 | TDR1 | R/W | H'FF | H'FF83 |
| | Serial status register 1 | SSR1 | R/(W)*[2] | H'84 | H'FF84 |
| | Receive data register 1 | RDR1 | R | H'00 | H'FF85 |
| | Smart card mode register 1 | SCMR1 | R/W | H'F2 | H'FF86 |
| All | Module stop control register | MSTPCR | R/W | H'3FFF | H'FF3C |

Notes: 1. Lower 16 bits of the address.
2. Can only be written with 0 for flag clearing.

## 12.2 Register Descriptions

### 12.2.1 Receive Shift Register (RSR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| R/W | : | — | — | — | — | — | — | — | — |

RSR is a register used to receive serial data.

The SCI sets serial data input from the RxD pin in RSR in the order received, starting with the LSB (bit 0), and converts it to parallel data. When one byte of data has been received, it is transferred to RDR automatically.

RSR cannot be directly read or written to by the CPU.

### 12.2.2 Receive Data Register (RDR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R | R | R | R | R | R | R | R |

RDR is a register that stores received serial data.

When the SCI has received one byte of serial data, it transfers the received serial data from RSR to RDR where it is stored, and completes the receive operation. After this, RSR is receive-enabled.

Since RSR and RDR function as a double buffer in this way, enables continuous receive operations to be performed.

RDR is a read-only register, and cannot be written to by the CPU.

RDR is initialized to H'00 by a reset, and in standby mode or module stop mode.

### 12.2.3 Transmit Shift Register (TSR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| R/W | : | — | — | — | — | — | — | — | — |

TSR is a register used to transmit serial data.

To perform serial data transmission, the SCI first transfers transmit data from TDR to TSR, then sends the data to the TxD pin starting with the LSB (bit 0).

When transmission of one byte is completed, the next transmit data is transferred from TDR to TSR, and transmission started, automatically. However, data transfer from TDR to TSR is not performed if the TDRE bit in SSR is set to 1.

TSR cannot be directly read or written to by the CPU.

### 12.2.4 Transmit Data Register (TDR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TDR is an 8-bit register that stores data for serial transmission.

When the SCI detects that TSR is empty, it transfers the transmit data written in TDR to TSR and starts serial transmission. Continuous serial transmission can be carried out by writing the next transmit data to TDR during serial transmission of the data in TSR.

TDR can be read or written to by the CPU at all times.

TDR is initialized to H'FF by a reset, and in standby mode or module stop mode.

### 12.2.5 Serial Mode Register (SMR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | C/$\overline{A}$ | CHR | PE | O/$\overline{E}$ | STOP | MP | CKS1 | CKS0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

SMR is an 8-bit register used to set the SCI's serial transfer format and select the baud rate generator clock source.

SMR can be read or written to by the CPU at all times.

SMR is initialized to H'00 by a reset, and in standby mode or module stop mode.

**Bit 7—Communication Mode (C/$\overline{A}$):** Selects asynchronous mode or clocked synchronous mode as the SCI operating mode.

**Bit 7**

| C/$\overline{A}$ | Description | |
|---|---|---|
| 0 | Asynchronous mode | (Initial value) |
| 1 | Clocked synchronous mode | |

**Bit 6—Character Length (CHR):** Selects 7 or 8 bits as the data length in asynchronous mode. In clocked synchronous mode, a fixed data length of 8 bits is used regardless of the CHR setting.

**Bit 6**

| CHR | Description | |
|---|---|---|
| 0 | 8-bit data | (Initial value) |
| 1 | 7-bit data* | |

Note: * When 7-bit data is selected, the MSB (bit 7) of TDR is not transmitted, and it is not possible to choose between LSB-first or MSB-first transfer.

**Bit 5—Parity Enable (PE):** In asynchronous mode, selects whether or not parity bit addition is performed in transmission, and parity bit checking in reception. In clocked synchronous mode, parity bit addition and checking is not performed, regardless of the PE bit setting.

**Bit 5**

| PE | Description | |
|---|---|---|
| 0 | Parity bit addition and checking disabled | (Initial value) |
| 1 | Parity bit addition and checking enabled* | |

Note:* When the PE bit is set to 1, the parity (even or odd) specified by the O/$\overline{E}$ bit is added to transmit data before transmission. In reception, the parity bit is checked for the parity (even or odd) specified by the O/$\overline{E}$ bit.

**Bit 4—Parity Mode (O/$\overline{E}$):** Selects either even or odd parity for use in parity addition and checking.

The O/$\overline{E}$ bit setting is only valid when the PE bit is set to 1, enabling parity bit addition and checking, in asynchronous mode. The O/$\overline{E}$ bit setting is invalid in clocked synchronous mode, and when parity addition and checking is disabled in asynchronous mode.

**Bit 4**

| O/$\overline{E}$ | Description | |
|---|---|---|
| 0 | Even parity*[1] | (Initial value) |
| 1 | Odd parity*[2] | |

Notes: 1. When even parity is set, parity bit addition is performed in transmission so that the total number of 1 bits in the transmit character plus the parity bit is even.
In reception, a check is performed to see if the total number of 1 bits in the receive character plus the parity bit is even.
2. When odd parity is set, parity bit addition is performed in transmission so that the total number of 1 bits in the transmit character plus the parity bit is odd.
In reception, a check is performed to see if the total number of 1 bits in the receive character plus the parity bit is odd.

**Bit 3—Stop Bit Length (STOP):** Selects 1 or 2 bits as the stop bit length in asynchronous mode. The STOP bits setting is only valid in asynchronous mode. If clocked synchronous mode is set the STOP bit setting is invalid since stop bits are not added.

| Bit 3 | |
|---|---|
| **STOP** | **Description** |
| 0 | 1 stop bit: In transmission, a single 1 bit (stop bit) is added to the end of each transmitted character before it is sent (Initial value) |
| 1 | 2 stop bits: In transmission, two 1 bits (stop bits) are added to the end of each transmitted character before it is sent |

In reception, only the first stop bit is checked, regardless of the STOP bit setting. If the second stop bit is 1, it is treated as a stop bit; if it is 0, it is treated as the start bit of the next transmit character.

**Bit 2—Multiprocessor Mode (MP):** Selects multiprocessor format. When multiprocessor format is selected, the PE bit and O/$\overline{E}$ bit parity settings are invalid. The MP bit setting is only valid in asynchronous mode; it is invalid in clocked synchronous mode.

For details of the multiprocessor communication function, see section 12.3.3, Multiprocessor Communication Function.

| Bit 2 | | |
|---|---|---|
| **MP** | **Description** | |
| 0 | Multiprocessor function disabled | (Initial value) |
| 1 | Multiprocessor format selected | |

**Bits 1 and 0—Clock Select 1 and 0 (CKS1, CKS0):** These bits select the clock source for the baud rate generator. The clock source can be selected from ø, ø/4, ø/16, and ø/64, according to the setting of bits CKS1 and CKS0.

For the relation between the clock source, the bit rate register setting, and the baud rate, see section 12.2.8, Bit Rate Register.

| Bit 1 | Bit 0 | | |
|---|---|---|---|
| **CKS1** | **CKS0** | **Description** | |
| 0 | 0 | ø clock | (Initial value) |
| | 1 | ø/4 clock | |
| 1 | 0 | ø/16 clock | |
| | 1 | ø/64 clock | |

### 12.2.6 Serial Control Register (SCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

SCR is a register that performs enabling or disabling of SCI transfer operations, serial clock output in asynchronous mode, and interrupt requests, and selection of the serial clock source.

SCR can be read or written to by the CPU at all times.

SCR is initialized to H'00 by a reset, and in standby mode or module stop mode.

**Bit 7—Transmit Interrupt Enable (TIE):** Enables or disables transmit data empty interrupt (TXI) request generation when serial transmit data is transferred from TDR to TSR and the TDRE flag in SSR is set to 1.

**Bit 7**

| TIE | Description | |
|---|---|---|
| 0 | Transmit data empty interrupt (TXI) requests disabled* | (Initial value) |
| 1 | Transmit data empty interrupt (TXI) requests enabled | |

Note:* TXI interrupt request cancellation can be performed by reading 1 from the TDRE flag, then clearing it to 0, or clearing the TIE bit to 0.

**Bit 6—Receive Interrupt Enable (RIE):** Enables or disables receive data full interrupt (RXI) request and receive error interrupt (ERI) request generation when serial receive data is transferred from RSR to RDR and the RDRF flag in SSR is set to 1.

**Bit 6**

| RIE | Description | |
|---|---|---|
| 0 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request disabled* | (Initial value) |
| 1 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request enabled | |

Note:* RXI and ERI interrupt request cancellation can be performed by reading 1 from the RDRF flag, or the FER, PER, or ORER flag, then clearing the flag to 0, or clearing the RIE bit to 0.

**Bit 5—Transmit Enable (TE):** Enables or disables the start of serial transmission by the SCI.

| Bit 5 | | |
|---|---|---|
| **TE** | **Description** | |
| 0 | Transmission disabled*[1] | (Initial value) |
| 1 | Transmission enabled*[2] | |

Notes: 1. The TDRE flag in SSR is fixed at 1.

2. In this state, serial transmission is started when transmit data is written to TDR and the TDRE flag in SSR is cleared to 0.

   SMR setting must be performed to decide the transfer format before setting the TE bit to 1.

**Bit 4—Receive Enable (RE):** Enables or disables the start of serial reception by the SCI.

| Bit 4 | | |
|---|---|---|
| **RE** | **Description** | |
| 0 | Reception disabled*[1] | (Initial value) |
| 1 | Reception enabled*[2] | |

Notes: 1. Clearing the RE bit to 0 does not affect the RDRF, FER, PER, and ORER flags, which retain their states.

2. Serial reception is started in this state when a start bit is detected in asynchronous mode or serial clock input is detected in clocked synchronous mode.

   SMR setting must be performed to decide the transfer format before setting the RE bit to 1.

**Bit 3—Multiprocessor Interrupt Enable (MPIE):** Enables or disables multiprocessor interrupts. The MPIE bit setting is only valid in asynchronous mode when the MP bit in SMR is set to 1.

The MPIE bit setting is invalid in clocked synchronous mode or when the MP bit is cleared to 0.

**Bit 3**

| MPIE | Description | |
|---|---|---|
| 0 | Multiprocessor interrupts disabled (normal reception performed)<br>[Clearing conditions]<br>• When the MPIE bit is cleared to 0<br>• When MPB= 1 data is received | (Initial value) |
| 1 | Multiprocessor interrupts enabled*<br>Receive interrupt (RXI) requests, receive error interrupt (ERI) requests, and setting of the RDRF, FER, and ORER flags in SSR are disabled until data with the multiprocessor bit set to 1 is received. | |

Note: * When receive data including MPB = 0 is received, receive data transfer from RSR to RDR, receive error detection, and setting of the RDRF, FER, and ORER flags in SSR , is not performed. When receive data including MPB = 1 is received, the MPB bit in SSR is set to 1, the MPIE bit is cleared to 0 automatically, and generation of RXI and ERI interrupts (when the TIE and RIE bits in SCR are set to 1) and FER and ORER flag setting is enabled.

**Bit 2—Transmit End Interrupt Enable (TEIE):** Enables or disables transmit end interrupt (TEI) request generation when there is no valid transmit data in TDR in MSB data transmission.

**Bit 2**

| TEIE | Description | |
|---|---|---|
| 0 | Transmit end interrupt (TEI) request disabled* | (Initial value) |
| 1 | Transmit end interrupt (TEI) request enabled* | |

Note: * TEI cancellation can be performed by reading 1 from the TDRE flag in SSR, then clearing it to 0 and clearing the TEND flag to 0, or clearing the TEIE bit to 0.

**Bits 1 and 0—Clock Enable 1 and 0 (CKE1, CKE0):** These bits are used to select the SCI clock source and enable or disable clock output from the SCK pin. The combination of the CKE1 and CKE0 bits determines whether the SCK pin functions as an I/O port, the serial clock output pin, or the serial clock input pin.

The setting of the CKE0 bit, however, is only valid for internal clock operation (CKE1 = 0) in asynchronous mode. The CKE0 bit setting is invalid in clocked synchronous mode, and in the case of external clock operation (CKE1 = 1). Note that the SCI's operating mode must be decided using SMR before setting the CKE1 and CKE0 bits.

For details of clock source selection, see table 12.9 in section 12.3, Operation.

| Bit 1 | Bit 0 | | |
|---|---|---|---|
| **CKE1** | **CKE0** | **Description** | |
| 0 | 0 | Asynchronous mode | Internal clock/SCK pin functions as I/O port*[1] |
| | | Clocked synchronous mode | Internal clock/SCK pin functions as serial clock output |
| | 1 | Asynchronous mode | Internal clock/SCK pin functions as clock output*[2] |
| | | Clocked synchronous mode | Internal clock/SCK pin functions as serial clock output |
| 1 | 0 | Asynchronous mode | External clock/SCK pin functions as clock input*[3] |
| | | Clocked synchronous mode | External clock/SCK pin functions as serial clock input |
| | 1 | Asynchronous mode | External clock/SCK pin functions as clock input*[3] |
| | | Clocked synchronous mode | External clock/SCK pin functions as serial clock input |

Notes: 1. Initial value
2. Outputs a clock of the same frequency as the bit rate.
3. Inputs a clock with a frequency 16 times the bit rate.

### 12.2.7 Serial Status Register (SSR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | TDRE | RDRF | ORER | FER | PER | TEND | MPB | MPBT |
| Initial value | : | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| R/W | : | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R | R | R/W |

Note: Only 0 can be written, to clear the flag.

SSR is an 8-bit register containing status flags that indicate the operating status of the SCI, and multiprocessor bits.

SSR can be read or written to by the CPU at all times. However, 1 cannot be written to flags TDRE, RDRF, ORER, PER, and FER. Also note that in order to clear these flags they must be read as 1 beforehand. The TEND flag and MPB flag are read-only flags and cannot be modified.

SSR is initialized to H'84 by a reset, and in standby mode or module stop mode.

**Bit 7—Transmit Data Register Empty (TDRE):** Indicates that data has been transferred from TDR to TSR and the next serial data can be written to TDR.

| Bit 7 TDRE | Description |
|---|---|
| 0 | [Clearing conditions]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DTC is activated by a TXI interrupt and writes data to TDR |
| 1 | [Setting conditions] (Initial value)<br>• When the TE bit in SCR is 0<br>• When data is transferred from TDR to TSR and data can be written to TDR |

**Bit 6—Receive Data Register Full (RDRF):** Indicates that the received data is stored in RDR.

| Bit 6 RDRF | Description |
|---|---|
| 0 | [Clearing conditions] (Initial value)<br>• When 0 is written to RDRF after reading RDRF = 1<br>• When the DTC is activated by an RXI interrupt and reads data from RDR |
| 1 | [Setting condition]<br>When serial reception ends normally and receive data is transferred from RSR to RDR |

Note: RDR and the RDRF flag are not affected and retain their previous values when an error is detected during reception or when the RE bit in SCR is cleared to 0.

If reception of the next data is completed while the RDRF flag is still set to 1, an overrun error will occur and the receive data will be lost.

**Bit 5—Overrun Error (ORER):** Indicates that an overrun error occurred during reception, causing abnormal termination.

**Bit 5**

| ORER | Description |
|---|---|
| 0 | [Clearing condition] (Initial value)*[1]<br>When 0 is written to ORER after reading ORER = 1 |
| 1 | [Setting condition]<br>When the next serial reception is completed while RDRF = 1 |

Notes: 1. The ORER flag is not affected and retains its previous state when the RE bit in SCR is cleared to 0.

2. The receive data prior to the overrun error is retained in RDR, and the data received subsequently is lost. Also, subsequent serial reception cannot be continued while the ORER flag is set to 1. In clocked synchronous mode, serial transmission cannot be continued, either.

**Bit 4—Framing Error (FER):** Indicates that a framing error occurred during reception in asynchronous mode, causing abnormal termination.

**Bit 4**

| FER | Description |
|---|---|
| 0 | [Clearing condition] (Initial value)*[1]<br>• When 0 is written to FER after reading FER = 1 |
| 1 | [Setting condition]<br>When the SCI checks whether the stop bit at the end of the receive data when reception ends, and the stop bit is 0 *[2] |

Notes: 1. The FER flag is not affected and retains its previous state when the RE bit in SCR is cleared to 0.

2. In 2-stop-bit mode, only the first stop bit is checked for a value of 0; the second stop bit is not checked. If a framing error occurs, the receive data is transferred to RDR but the RDRF flag is not set. Also, subsequent serial reception cannot be continued while the FER flag is set to 1. In clocked synchronous mode, serial transmission cannot be continued, either.

**Bit 3—Parity Error (PER):** Indicates that a parity error occurred during reception using parity addition in asynchronous mode, causing abnormal termination.

**Bit 3**

| PER | Description |
|---|---|
| 0 | [Clearing condition] (Initial value)*[1]<br>When 0 is written to PER after reading PER = 1 |
| 1 | [Setting condition]<br>When, in reception, the number of 1 bits in the receive data plus the parity bit does not match the parity setting (even or odd) specified by the O/$\overline{E}$ bit in SMR*[2] |

Notes: 1. The PER flag is not affected and retains its previous state when the RE bit in SCR is cleared to 0.
2. If a parity error occurs, the receive data is transferred to RDR but the RDRF flag is not set. Also, subsequent serial reception cannot be continued while the PER flag is set to 1. In clocked synchronous mode, serial transmission cannot be continued, either.

**Bit 2—Transmit End (TEND):** Indicates that there is no valid data in TDR when the last bit of the transmit character is sent, and transmission has been ended.

The TEND flag is read-only and cannot be modified.

**Bit 2**

| TEND | Description |
|---|---|
| 0 | [Clearing conditions]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DTC is activated by a TXI interrupt and writes data to TDR |
| 1 | [Setting conditions] (Initial value)<br>• When the TE bit in SCR is 0<br>• When TDRE = 1 at transmission of the last bit of a 1-byte serial transmit character |

**Bit 1—Multiprocessor Bit (MPB):** When reception is performed using multiprocessor format in asynchronous mode, MPB stores the multiprocessor bit in the receive data.

MPB is a read-only bit, and cannot be modified.

**Bit 1**

| MPB | Description |
|---|---|
| 0 | [Clearing condition] (Initial value)*<br>When data with a 0 multiprocessor bit is received |
| 1 | [Setting condition]<br>When data with a 1 multiprocessor bit is received |

Note: * Retains its previous state when the RE bit in SCR is cleared to 0 with multiprocessor format.

**Bit 0—Multiprocessor Bit Transfer (MPBT):** When transmission is performed using multiprocessor format in asynchronous mode, MPBT stores the multiprocessor bit to be added to the transmit data.

The MPBT bit setting is invalid when multiprocessor format is not used, when not transmitting, and in clocked synchronous mode.

**Bit 0**

| MPBT | Description | |
|---|---|---|
| 0 | Data with a 0 multiprocessor bit is transmitted | (Initial value) |
| 1 | Data with a 1 multiprocessor bit is transmitted | |

### 12.2.8 Bit Rate Register (BRR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

BRR is an 8-bit register that sets the serial transfer bit rate in accordance with the baud rate generator operating clock selected by bits CKS1 and CKS0 in SMR.

BRR can be read or written to by the CPU at all times.

BRR is initialized to H'FF by a reset, and in standby mode or module stop mode.

As baud rate generator control is performed independently for each channel, different values can be set for each channel.

Table 12.3 shows sample BRR settings in asynchronous mode, and table 12.4 shows sample BRR settings in clocked synchronous mode.

**Table 12.3 BRR Settings for Various Bit Rates (Asynchronous Mode)**

| Bit Rate (bit/s) | ø = 2 MHz | | | ø = 2.097152 MHz | | | ø = 2.4576 MHz | | | ø = 3 MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 1 | 141 | 0.03 | 1 | 148 | –0.04 | 1 | 174 | –0.26 | 1 | 212 | 0.03 |
| 150 | 1 | 103 | 0.16 | 1 | 108 | 0.21 | 1 | 127 | 0.00 | 1 | 155 | 0.16 |
| 300 | 0 | 207 | 0.16 | 0 | 217 | 0.21 | 0 | 255 | 0.00 | 1 | 77 | 0.16 |
| 600 | 0 | 103 | 0.16 | 0 | 108 | 0.21 | 0 | 127 | 0.00 | 0 | 155 | 0.16 |
| 1200 | 0 | 51 | 0.16 | 0 | 54 | –0.70 | 0 | 63 | 0.00 | 0 | 77 | 0.16 |
| 2400 | 0 | 25 | 0.16 | 0 | 26 | 1.14 | 0 | 31 | 0.00 | 0 | 38 | 0.16 |
| 4800 | 0 | 12 | 0.16 | 0 | 13 | –2.48 | 0 | 15 | 0.00 | 0 | 19 | –2.34 |
| 9600 | 0 | 6 | — | 0 | 6 | –2.48 | 0 | 7 | 0.00 | 0 | 9 | –2.34 |
| 19200 | 0 | 2 | — | 0 | 2 | — | 0 | 3 | 0.00 | 0 | 4 | –2.34 |
| 31250 | 0 | 1 | 0.00 | 0 | 1 | — | 0 | 1 | — | 0 | 2 | 0.00 |
| 38400 | 0 | 1 | — | 0 | 1 | — | 0 | 1 | 0.00 | — | — | — |

| Bit Rate (bit/s) | ø = 3.6864 MHz | | | ø = 4 MHz | | | ø = 4.9152 MHz | | | ø = 5 MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 2 | 64 | 0.70 | 2 | 70 | 0.03 | 2 | 86 | 0.31 | 2 | 88 | –0.25 |
| 150 | 1 | 191 | 0.00 | 1 | 207 | 0.16 | 1 | 255 | 0.00 | 2 | 64 | 0.16 |
| 300 | 1 | 95 | 0.00 | 1 | 103 | 0.16 | 1 | 127 | 0.00 | 1 | 129 | 0.16 |
| 600 | 0 | 191 | 0.00 | 0 | 207 | 0.16 | 0 | 255 | 0.00 | 1 | 64 | 0.16 |
| 1200 | 0 | 95 | 0.00 | 0 | 103 | 0.16 | 0 | 127 | 0.00 | 0 | 129 | 0.16 |
| 2400 | 0 | 47 | 0.00 | 0 | 51 | 0.16 | 0 | 63 | 0.00 | 0 | 64 | 0.16 |
| 4800 | 0 | 23 | 0.00 | 0 | 25 | 0.16 | 0 | 31 | 0.00 | 0 | 32 | –1.36 |
| 9600 | 0 | 11 | 0.00 | 0 | 12 | 0.16 | 0 | 15 | 0.00 | 0 | 15 | 1.73 |
| 19200 | 0 | 5 | 0.00 | 0 | 6 | — | 0 | 7 | 0.00 | 0 | 7 | 1.73 |
| 31250 | — | — | — | 0 | 3 | 0.00 | 0 | 4 | –1.70 | 0 | 4 | 0.00 |
| 38400 | 0 | 2 | 0.00 | 0 | 2 | — | 0 | 3 | 0.00 | 0 | 3 | 1.73 |

Note: Settings with an error of 1% or less are recommended.

**Legend**

—: Setting possible, but error occurs

**Table 12.3 BRR Settings for Various Bit Rates (Asynchronous Mode) (cont)**

| | ø = 6 MHz | | | ø = 6.144 MHz | | | ø = 7.3728 MHz | | | ø = 8 MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bit Rate (bit/s)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** |
| 110 | 2 | 106 | –0.44 | 2 | 108 | 0.08 | 2 | 130 | –0.07 | 2 | 141 | 0.03 |
| 150 | 2 | 77 | 0.16 | 2 | 79 | 0.00 | 2 | 95 | 0.00 | 2 | 103 | 0.16 |
| 300 | 1 | 155 | 0.16 | 1 | 159 | 0.00 | 1 | 191 | 0.00 | 1 | 207 | 0.16 |
| 600 | 1 | 77 | 0.16 | 1 | 79 | 0.00 | 1 | 95 | 0.00 | 1 | 103 | 0.16 |
| 1200 | 0 | 155 | 0.16 | 0 | 159 | 0.00 | 0 | 191 | 0.00 | 0 | 207 | 0.16 |
| 2400 | 0 | 77 | 0.16 | 0 | 79 | 0.00 | 0 | 95 | 0.00 | 0 | 103 | 0.16 |
| 4800 | 0 | 38 | 0.16 | 0 | 39 | 0.00 | 0 | 47 | 0.00 | 0 | 51 | 0.16 |
| 9600 | 0 | 19 | –2.34 | 0 | 19 | 0.00 | 0 | 23 | 0.00 | 0 | 25 | 0.16 |
| 19200 | 0 | 9 | –2.34 | 0 | 9 | 0.00 | 0 | 11 | 0.00 | 0 | 12 | 0.16 |
| 31250 | 0 | 5 | 0.00 | 0 | 5 | 2.40 | 0 | 6 | — | 0 | 7 | 0.00 |
| 38400 | 0 | 4 | –2.34 | 0 | 4 | 0.00 | 0 | 5 | 0.00 | 0 | 6 | — |

| | ø = 9.8304 MHz | | | ø = 10 MHz | | | ø = 12 MHz | | | ø = 12.288 MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bit Rate (bit/s)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** |
| 110 | 2 | 174 | –0.26 | 2 | 177 | –0.25 | 2 | 212 | 0.03 | 2 | 217 | 0.08 |
| 150 | 2 | 127 | 0.00 | 2 | 129 | 0.16 | 2 | 155 | 0.16 | 2 | 159 | 0.00 |
| 300 | 1 | 255 | 0.00 | 2 | 64 | 0.16 | 2 | 77 | 0.16 | 2 | 79 | 0.00 |
| 600 | 1 | 127 | 0.00 | 1 | 129 | 0.16 | 1 | 155 | 0.16 | 1 | 159 | 0.00 |
| 1200 | 0 | 255 | 0.00 | 1 | 64 | 0.16 | 1 | 77 | 0.16 | 1 | 79 | 0.00 |
| 2400 | 0 | 127 | 0.00 | 0 | 129 | 0.16 | 0 | 155 | 0.16 | 0 | 159 | 0.00 |
| 4800 | 0 | 63 | 0.00 | 0 | 64 | 0.16 | 0 | 77 | 0.16 | 0 | 79 | 0.00 |
| 9600 | 0 | 31 | 0.00 | 0 | 32 | –1.36 | 0 | 38 | 0.16 | 0 | 39 | 0.00 |
| 19200 | 0 | 15 | 0.00 | 0 | 15 | 1.73 | 0 | 19 | –2.34 | 0 | 19 | 0.00 |
| 31250 | 0 | 9 | –1.70 | 0 | 9 | 0.00 | 0 | 11 | 0.00 | 0 | 11 | 2.40 |
| 38400 | 0 | 7 | 0.00 | 0 | 7 | 1.73 | 0 | 9 | –2.34 | 0 | 9 | 0.00 |

Note: Settings with an error of 1% or less are recommended.

**Legend**

—: Setting possible, but error occurs

**Table 12.3 BRR Settings for Various Bit Rates (Asynchronous Mode) (cont)**

| Bit Rate (bit/s) | ø = 14 MHz | | | ø = 14.7456 MHz | | | ø = 16 MHz | | | ø = 17.2032 MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 2 | 248 | –0.17 | 3 | 64 | 0.70 | 3 | 70 | 0.03 | 3 | 75 | 0.48 |
| 150 | 2 | 181 | 0.16 | 2 | 191 | 0.00 | 2 | 207 | 0.16 | 2 | 223 | 0.00 |
| 300 | 2 | 90 | 0.16 | 2 | 95 | 0.00 | 2 | 103 | 0.16 | 2 | 111 | 0.00 |
| 600 | 1 | 181 | 0.16 | 1 | 191 | 0.00 | 1 | 207 | 0.16 | 1 | 223 | 0.00 |
| 1200 | 1 | 90 | 0.16 | 1 | 95 | 0.00 | 1 | 103 | 0.16 | 1 | 111 | 0.00 |
| 2400 | 0 | 181 | 0.16 | 0 | 191 | 0.00 | 0 | 207 | 0.16 | 0 | 223 | 0.00 |
| 4800 | 0 | 90 | 0.16 | 0 | 95 | 0.00 | 0 | 103 | 0.16 | 0 | 111 | 0.00 |
| 9600 | 0 | 45 | –0.93 | 0 | 47 | 0.00 | 0 | 51 | 0.16 | 0 | 55 | 0.00 |
| 19200 | 0 | 22 | –0.93 | 0 | 23 | 0.00 | 0 | 25 | 0.16 | 0 | 27 | 0.00 |
| 31250 | 0 | 13 | 0.00 | 0 | 14 | –1.70 | 0 | 15 | 0.00 | 0 | 16 | 1.20 |
| 38400 | 0 | 10 | — | 0 | 11 | 0.00 | 0 | 12 | 0.16 | 0 | 13 | 0.00 |

| Bit Rate (bit/s) | ø = 18 MHz | | | ø = 19.6608 MHz | | | ø = 20 MHz | | |
|---|---|---|---|---|---|---|---|---|---|
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 3 | 79 | –0.12 | 3 | 86 | 0.31 | 3 | 88 | –0.25 |
| 150 | 2 | 233 | 0.16 | 2 | 255 | 0.00 | 3 | 64 | 0.16 |
| 300 | 2 | 116 | 0.16 | 2 | 127 | 0.00 | 2 | 129 | 0.16 |
| 600 | 1 | 233 | 0.16 | 1 | 255 | 0.00 | 2 | 64 | 0.16 |
| 1200 | 1 | 116 | 0.16 | 1 | 127 | 0.00 | 1 | 129 | 0.16 |
| 2400 | 0 | 233 | 0.16 | 0 | 255 | 0.00 | 1 | 64 | 0.16 |
| 4800 | 0 | 116 | 0.16 | 0 | 127 | 0.00 | 0 | 129 | 0.16 |
| 9600 | 0 | 58 | –0.69 | 0 | 63 | 0.00 | 0 | 64 | 0.16 |
| 19200 | 0 | 28 | 1.02 | 0 | 31 | 0.00 | 0 | 32 | –1.36 |
| 31250 | 0 | 17 | 0.00 | 0 | 19 | –1.70 | 0 | 19 | 0.00 |
| 38400 | 0 | 14 | –2.34 | 0 | 15 | 0.00 | 0 | 15 | 1.73 |

Note: Settings with an error of 1% or less are recommended.

**Legend**

—: Setting possible, but error occurs

**Table 12.4 BRR Settings for Various Bit Rates (Clocked Synchronous Mode)**

| Bit Rate | ø = 2 MHz | | ø = 4 MHz | | ø = 8 MHz | | ø = 10 MHz | | ø = 16 MHz | | ø = 20 MHz | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| (bit/s) | n | N | n | N | n | N | n | N | n | N | n | N |
| 110 | 3 | 70 | — | — | | | | | | | | |
| 250 | 2 | 124 | 2 | 249 | 3 | 124 | — | — | 3 | 249 | | |
| 500 | 1 | 249 | 2 | 124 | 2 | 249 | — | — | 3 | 124 | — | — |
| 1 k | 1 | 124 | 1 | 249 | 2 | 124 | — | — | 2 | 249 | — | — |
| 2.5 k | 0 | 199 | 1 | 99 | 1 | 199 | 1 | 249 | 2 | 99 | 2 | 124 |
| 5 k | 0 | 99 | 0 | 199 | 1 | 99 | 1 | 124 | 1 | 199 | 1 | 249 |
| 10 k | 0 | 49 | 0 | 99 | 0 | 199 | 0 | 249 | 1 | 99 | 1 | 124 |
| 25 k | 0 | 19 | 0 | 39 | 0 | 79 | 0 | 99 | 0 | 159 | 0 | 199 |
| 50 k | 0 | 9 | 0 | 19 | 0 | 39 | 0 | 49 | 0 | 79 | 0 | 99 |
| 100 k | 0 | 4 | 0 | 9 | 0 | 19 | 0 | 24 | 0 | 39 | 0 | 49 |
| 250 k | 0 | 1 | 0 | 3 | 0 | 7 | 0 | 9 | 0 | 15 | 0 | 19 |
| 500 k | 0 | 0* | 0 | 1 | 0 | 3 | 0 | 4 | 0 | 7 | 0 | 9 |
| 1 M | | | 0 | 0* | 0 | 1 | — | — | 0 | 3 | 0 | 4 |
| 2.5 M | | | | | — | — | 0 | 0* | — | — | 0 | 1 |
| 5 M | | | | | | | | | — | — | 0 | 0* |

**Legend**

Blank : Cannot be set.

— : Can be set, but there will be a degree of error.

* : Continuous transfer is not possible.

The BRR setting is found from the following formulas.

Asynchronous mode:

$$N = \frac{\phi}{64 \times 2^{2n-1} \times B} \times 10^6 - 1$$

Clocked synchronous mode:

$$N = \frac{\phi}{8 \times 2^{2n-1} \times B} \times 10^6 - 1$$

Where B: Bit rate (bit/s)
N: BRR setting for baud rate generator ($0 \leq N \leq 255$)
ø: Operating frequency (MHz)
n: Baud rate generator input clock (n = 0 to 3)
(See the table below for the relation between n and the clock.)

| | | SMR Setting | |
|---|---|---|---|
| **n** | **Clock** | **CKS1** | **CKS0** |
| 0 | ø | 0 | 0 |
| 1 | ø/4 | 0 | 1 |
| 2 | ø/16 | 1 | 0 |
| 3 | ø/64 | 1 | 1 |

The bit rate error in asynchronous mode is found from the following formula:

$$\text{Error (\%)} = \left\{ \frac{\phi \times 10^6}{(N + 1) \times B \times 64 \times 2^{2n-1}} - 1 \right\} \times 100$$

Table 12.5 shows the maximum bit rate for each frequency in asynchronous mode. Tables 12.6 and 12.7 show the maximum bit rates with external clock input.

**Table 12.5 Maximum Bit Rate for Each Frequency (Asynchronous Mode)**

| ø (MHz) | Maximum Bit Rate (bit/s) | n | N |
|---|---|---|---|
| 2 | 62500 | 0 | 0 |
| 2.097152 | 65536 | 0 | 0 |
| 2.4576 | 76800 | 0 | 0 |
| 3 | 93750 | 0 | 0 |
| 3.6864 | 115200 | 0 | 0 |
| 4 | 125000 | 0 | 0 |
| 4.9152 | 153600 | 0 | 0 |
| 5 | 156250 | 0 | 0 |
| 6 | 187500 | 0 | 0 |
| 6.144 | 192000 | 0 | 0 |
| 7.3728 | 230400 | 0 | 0 |
| 8 | 250000 | 0 | 0 |
| 9.8304 | 307200 | 0 | 0 |
| 10 | 312500 | 0 | 0 |
| 12 | 375000 | 0 | 0 |
| 12.288 | 384000 | 0 | 0 |
| 14 | 437500 | 0 | 0 |
| 14.7456 | 460800 | 0 | 0 |
| 16 | 500000 | 0 | 0 |
| 17.2032 | 537600 | 0 | 0 |
| 18 | 562500 | 0 | 0 |
| 19.6608 | 614400 | 0 | 0 |
| 20 | 625000 | 0 | 0 |

**Table 12.6 Maximum Bit Rate with External Clock Input (Asynchronous Mode)**

| ø (MHz) | External Input Clock (MHz) | Maximum Bit Rate (bit/s) |
|---|---|---|
| 2 | 0.5000 | 31250 |
| 2.097152 | 0.5243 | 32768 |
| 2.4576 | 0.6144 | 38400 |
| 3 | 0.7500 | 46875 |
| 3.6864 | 0.9216 | 57600 |
| 4 | 1.0000 | 62500 |
| 4.9152 | 1.2288 | 76800 |
| 5 | 1.2500 | 78125 |
| 6 | 1.5000 | 93750 |
| 6.144 | 1.5360 | 96000 |
| 7.3728 | 1.8432 | 115200 |
| 8 | 2.0000 | 125000 |
| 9.8304 | 2.4576 | 153600 |
| 10 | 2.5000 | 156250 |
| 12 | 3.0000 | 187500 |
| 12.288 | 3.0720 | 192000 |
| 14 | 3.5000 | 218750 |
| 14.7456 | 3.6864 | 230400 |
| 16 | 4.0000 | 250000 |
| 17.2032 | 4.3008 | 268800 |
| 18 | 4.5000 | 281250 |
| 19.6608 | 4.9152 | 307200 |
| 20 | 5.0000 | 312500 |

**Table 12.7 Maximum Bit Rate with External Clock Input (Clocked Synchronous Mode)**

| ø (MHz) | External Input Clock (MHz) | Maximum Bit Rate (bit/s) |
|---|---|---|
| 2 | 0.3333 | 333333.3 |
| 4 | 0.6667 | 666666.7 |
| 6 | 1.0000 | 1000000.0 |
| 8 | 1.3333 | 1333333.3 |
| 10 | 1.6667 | 1666666.7 |
| 12 | 2.0000 | 2000000.0 |
| 14 | 2.3333 | 2333333.3 |
| 16 | 2.6667 | 2666666.7 |
| 18 | 3.0000 | 3000000.0 |
| 20 | 3.3333 | 3333333.3 |

### 12.2.9 Smart Card Mode Register (SCMR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | SDIR | SINV | — | SMIF |
| Initial value | : | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| R/W | : | — | — | — | — | R/W | R/W | — | R/W |

SCMR selects LSB-first or MSB-first by means of bit SDIR. Except in the case of asynchronous mode 7-bit data, LSB-first or MSB-first can be selected regardless of the serial communication mode. The descriptions in this chapter refer to LSB-first transfer.

For details of the other bits in SCMR, see 13.2.1, Smart Card Mode Register (SCMR).

SCMR is initialized to H'F2 by a reset, and in standby mode or module stop mode.

**Bits 7 to 4—Reserved:** Read-only bits, always read as 1.

**Bit 3—Smart Card Data Transfer Direction (SDIR):** Selects the serial/parallel conversion format.

The transmit/receive format is valid for 8-bit data.

**Bit 3**

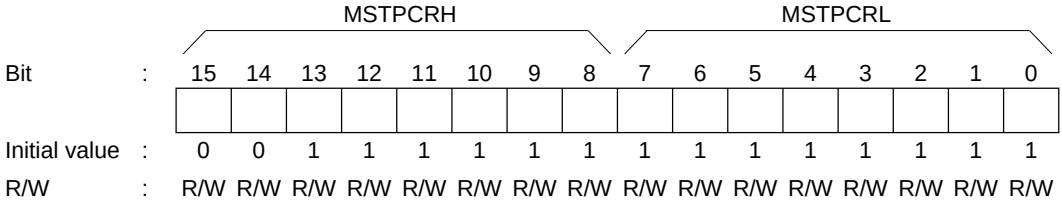
| SDIR | Description | |
|---|---|---|
| 0 | TDR contents are transmitted LSB-first | (Initial value) |
| | Receive data is stored in RDR LSB-first | |
| 1 | TDR contents are transmitted MSB-first | |
| | Receive data is stored in RDR MSB-first | |

**Bit 2—Smart Card Data Invert (SINV):** When the smart card interface operates as a normal SCI, 0 should be written in this bit.

**Bit 1—Reserved:** Read-only bit, always read as 1.

**Bit 0—Smart Card Interface Mode Select (SMIF):** When the smart card interface operates as a normal SCI, 0 should be written in this bit.

### 12.2.10 Module Stop Control Register (MSTPCR)




MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the corresponding bit of bits MSTP6 to MSTP5 is set to 1, SCI operation stops at the end of the bus cycle and a transition is made to module stop mode. Registers cannot be read or written to in module stop mode. For details, see section 19.5, Module Stop Mode.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 6—Module Stop (MSTP6):** Specifies the SCI channel 1 module stop mode.

| Bit 6 | | |
|---|---|---|
| **MSTP6** | **Description** | |
| 0 | SCI channel 1 module stop mode cleared | |
| 1 | SCI channel 1 module stop mode set | (Initial value) |

**Bit 5—Module Stop (MSTP5):** Specifies the SCI channel 0 module stop mode.

| Bit 5 | | |
|---|---|---|
| **MSTP5** | **Description** | |
| 0 | SCI channel 0 module stop mode cleared | |
| 1 | SCI channel 0 module stop mode set | (Initial value) |

## 12.3 Operation

### 12.3.1 Overview

The SCI can carry out serial communication in two modes: asynchronous mode in which synchronization is achieved character by character, and clocked synchronous mode in which synchronization is achieved with clock pulses.

Selection of asynchronous or clocked synchronous mode and the transmission format is made using SMR as shown in table 12.8. The SCI clock is determined by a combination of the C/$\overline{A}$ bit in SMR and the CKE1 and CKE0 bits in SCR, as shown in table 12.9.

**Asynchronous Mode**

- Data length: Choice of 7 or 8 bits
- Choice of parity addition, multiprocessor bit addition, and addition of 1 or 2 stop bits (the combination of these parameters determines the transfer format and character length)
- Detection of framing, parity, and overrun errors, and breaks, during reception
- Choice of internal or external clock as SCI clock source
  - — When internal clock is selected:

    The SCI operates on the baud rate generator clock and a clock with the same frequency as the bit rate can be output
  - — When external clock is selected:

    A clock with a frequency of 16 times the bit rate must be input (the on-chip baud rate generator is not used)

**Clocked Synchronous Mode**

- Transfer format: Fixed 8-bit data
- Detection of overrun errors during reception
- Choice of internal or external clock as SCI clock source
  - — When internal clock is selected:

    The SCI operates on the baud rate generator clock and a serial clock is output off-chip
  - — When external clock is selected:

    The on-chip baud rate generator is not used, and the SCI operates on the input serial clock

**Table 12.8 SMR Settings and Serial Transfer Format Selection**

| SMR Settings | | | | | | SCI Transfer Format | | | |
|---|---|---|---|---|---|---|---|---|---|
| Bit 7 | Bit 6 | Bit 2 | Bit 5 | Bit 3 | | Data | Multi-processor | Parity | Stop Bit |
| C/$\overline{A}$ | CHR | MP | PE | STOP | Mode | Length | Bit | Bit | Length |
| 0 | 0 | 0 | 0 | 0 | Asynchronous mode | 8-bit data | No | No | 1 bit |
| | | | | 1 | | | | | 2 bits |
| | | | 1 | 0 | | | | Yes | 1 bit |
| | | | | 1 | | | | | 2 bits |
| | 1 | | 0 | 0 | | 7-bit data | | No | 1 bit |
| | | | | 1 | | | | | 2 bits |
| | | | 1 | 0 | | | | Yes | 1 bit |
| | | | | 1 | | | | | 2 bits |
| | 0 | 1 | — | 0 | Asynchronous mode (multi-processor format) | 8-bit data | Yes | No | 1 bit |
| | | | — | 1 | | | | | 2 bits |
| | 1 | | — | 0 | | 7-bit data | | | 1 bit |
| | | | — | 1 | | | | | 2 bits |
| 1 | — | — | — | — | Clocked synchronous mode | 8-bit data | No | | None |

**Table 12.9 SMR and SCR Settings and SCI Clock Source Selection**

| SMR | SCR Setting | | | SCI Transmit/Receive Clock | |
|---|---|---|---|---|---|
| Bit 7 | Bit 1 | Bit 0 | | Clock | |
| C/$\overline{A}$ | CKE1 | CKE0 | Mode | Source | SCK Pin Function |
| 0 | 0 | 0 | Asynchronous mode | Internal | SCI does not use SCK pin |
| | | 1 | | | Outputs clock with same frequency as bit rate |
| | 1 | 0 | | External | Inputs clock with frequency of 16 times the bit rate |
| | | 1 | | | |
| 1 | 0 | 0 | Clocked synchronous mode | Internal | Outputs serial clock |
| | | 1 | | | |
| | 1 | 0 | | External | Inputs serial clock |
| | | 1 | | | |

### 12.3.2 Operation in Asynchronous Mode

In asynchronous mode, characters are sent or received, each preceded by a start bit indicating the start of communication and one or two stop bits indicating the end of communication. Serial communication is thus carried out with synchronization established on a character-by-character basis.

Inside the SCI, the transmitter and receiver are independent units, enabling full-duplex communication. Both the transmitter and the receiver also have a double-buffered structure, so that data can be read or written during transmission or reception, enabling continuous data transfer.

Figure 12.2 shows the general format for asynchronous serial communication.

In asynchronous serial communication, the transmission line is usually held in the mark state (high level). The SCI monitors the transmission line, and when it goes to the space state (low level), recognizes a start bit and starts serial communication.

One serial communication character consists of a start bit (low level), followed by data (in LSB-first order), a parity bit (high or low level), and finally one or two stop bits (high level).

In asynchronous mode, the SCI performs synchronization at the falling edge of the start bit in reception. The SCI samples the data on the 8th pulse of a clock with a frequency of 16 times the length of one bit, so that the transfer data is latched at the center of each bit.

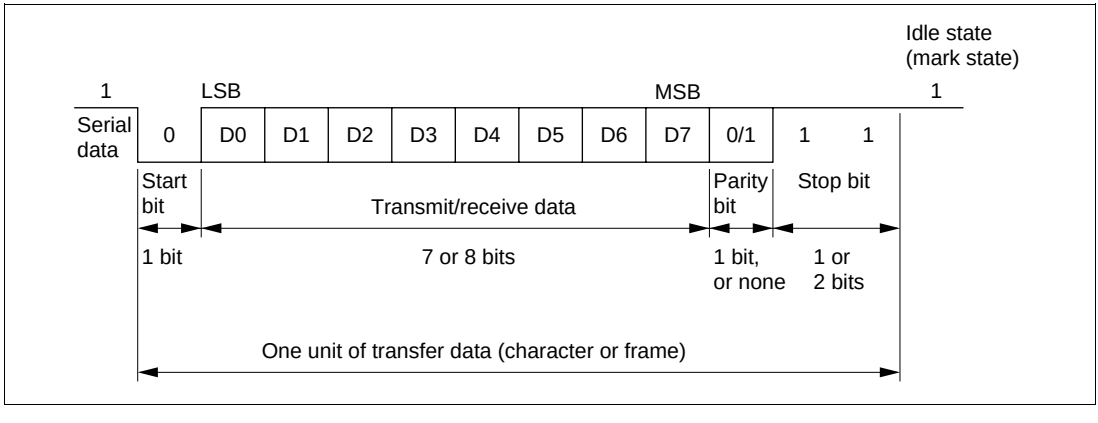



**Figure 12.2 Data Format in Asynchronous Communication (Example with 8-Bit Data, Parity, Two Stop Bits)**

**Data Transfer Format:** Table 12.10 shows the data transfer formats that can be used in asynchronous mode. Any of 12 transfer formats can be selected according to the SMR setting.

**Table 12.10 Serial Transfer Formats (Asynchronous Mode)**

| SMR Settings | | | | Serial Transfer Format and Frame Length |
|---|---|---|---|---|
| CHR | PE | MP | STOP | 1 2 3 4 5 6 7 8 9 10 11 12 |
| 0 | 0 | 0 | 0 | S 8-bit data STOP |
| 0 | 0 | 0 | 1 | S 8-bit data STOP STOP |
| 0 | 1 | 0 | 0 | S 8-bit data P STOP |
| 0 | 1 | 0 | 1 | S 8-bit data P STOP STOP |
| 1 | 0 | 0 | 0 | S 7-bit data STOP |
| 1 | 0 | 0 | 1 | S 7-bit data STOP STOP |
| 1 | 1 | 0 | 0 | S 7-bit data P STOP |
| 1 | 1 | 0 | 1 | S 7-bit data P STOP STOP |
| 0 | — | 1 | 0 | S 8-bit data MPB STOP |
| 0 | — | 1 | 1 | S 8-bit data MPB STOP STOP |
| 1 | — | 1 | 0 | S 7-bit data MPB STOP |
| 1 | — | 1 | 1 | S 7-bit data MPB STOP STOP |

**Legend**

S : Start bit
STOP : Stop bit
P : Parity bit
MPB : Multiprocessor bit

**Clock:** Either an internal clock generated by the on-chip baud rate generator or an external clock input at the SCK pin can be selected as the SCI's serial clock, according to the setting of the C/$\overline{A}$ bit in SMR and the CKE1 and CKE0 bits in SCR. For details of SCI clock source selection, see table 12.9.

When an external clock is input at the SCK pin, the clock frequency should be 16 times the bit rate used.

When the SCI is operated on an internal clock, the clock can be output from the SCK pin. The frequency of the clock output in this case is equal to the bit rate, and the phase is such that the rising edge of the clock is in the middle of the transmit data, as shown in figure 12.3.

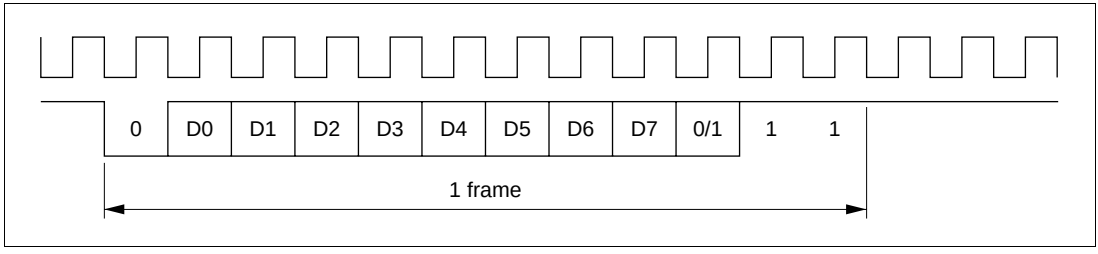



**Figure 12.3 Relation between Output Clock and Transfer Data Phase (Asynchronous Mode)**

**Data Transfer Operations:**

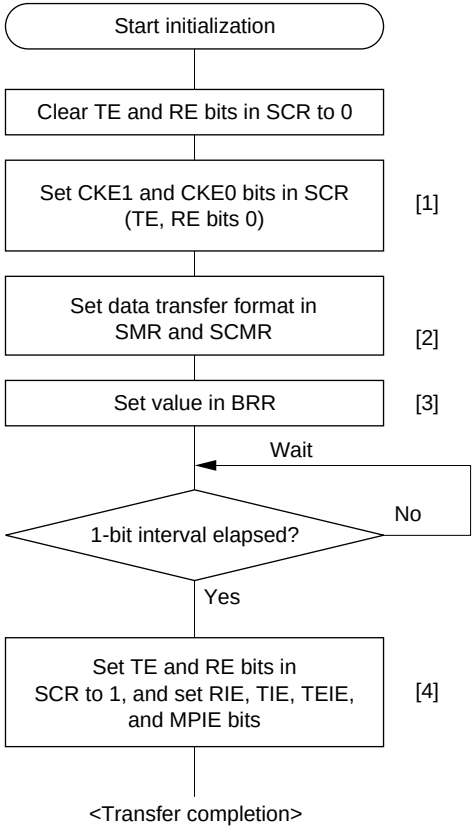
- SCI initialization (asynchronous mode)

  Before transmitting and receiving data, you should first clear the TE and RE bits in SCR to 0, then initialize the SCI as described below.

  When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change using the following procedure. When the TE bit is cleared to 0, the TDRE flag is set to 1 and TSR is initialized. Note that clearing the RE bit to 0 does not change the contents of the RDRF, PER, FER, and ORER flags, or the contents of RDR.

  When an external clock is used the clock should not be stopped during operation, including initialization, since operation is uncertain.

Figure 12.4 shows a sample SCI initialization flowchart.




[1] Set the clock selection in SCR. Be sure to clear bits RIE, TIE, TEIE, and MPIE, and bits TE and RE, to 0.

When the clock is selected in asynchronous mode, it is output immediately after SCR settings are made.

[2] Set the data transfer format in SMR and SCMR.

[3] Write a value corresponding to the bit rate to BRR. Not necessary if an external clock is used.

[4] Wait at least one bit interval, then set the TE bit or RE bit in SCR to 1. Also set the RIE, TIE, TEIE, and MPIE bits.

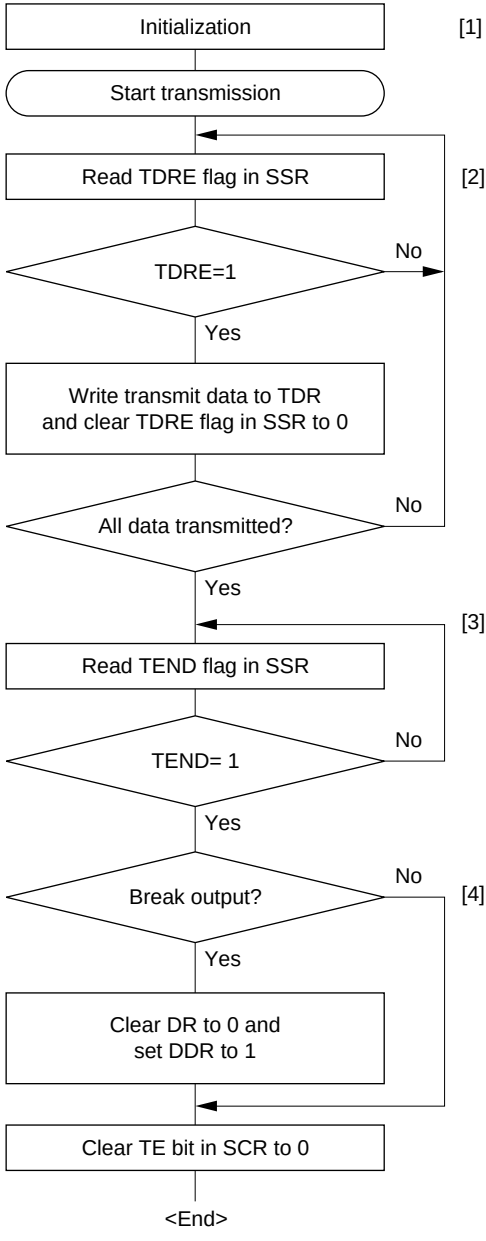
Setting the TE and RE bits enables the TxD and RxD pins to be used.

**Figure 12.4 Sample SCI Initialization Flowchart**

- Serial data transmission (asynchronous mode)

  Figure 12.5 shows a sample flowchart for serial transmission.

  The following procedure should be used for serial data transmission.




[1] SCI initialization:
The TxD pin is automatically designated as the transmit data output pin.
After the TE bit is set to 1, a frame of 1s is output, and transmission is enabled.

[2] SCI status check and transmit data write:
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR and clear the TDRE flag to 0.

[3] Serial transmission continuation procedure:
To continue serial transmission, read 1 from the TDRE flag to confirm that writing is possible, then write data to TDR, and then clear the TDRE flag to 0. Checking and clearing of the TDRE flag is automatic when the DTC is activated by a transmit data empty interrupt (TXI) request, and date is written to TDR.

[4] Break output at the end of serial transmission:
To output a break in serial transmission, set DDR for the port corresponding to the TxD pin to 1, clear DR to 0, then clear the TE bit in SCR to 0.

**Figure 12.5 Sample Serial Transmission Flowchart**

In serial transmission, the SCI operates as described below.

[1] The SCI monitors the TDRE flag in SSR, and if is 0, recognizes that data has been written to TDR, and transfers the data from TDR to TSR.

[2] After transferring data from TDR to TSR, the SCI sets the TDRE flag to 1 and starts transmission.

If the TIE bit is set to 1 at this time, a transmit data empty interrupt (TXI) is generated.

The serial transmit data is sent from the TxD pin in the following order.

[a] Start bit:

One 0-bit is output.

[b] Transmit data:

8-bit or 7-bit data is output in LSB-first order.

[c] Parity bit or multiprocessor bit:

One parity bit (even or odd parity), or one multiprocessor bit is output.

A format in which neither a parity bit nor a multiprocessor bit is output can also be selected.

[d] Stop bit(s):

One or two 1-bits (stop bits) are output.

[e] Mark state:

1 is output continuously until the start bit that starts the next transmission is sent.

[3] The SCI checks the TDRE flag at the timing for sending the stop bit.

If the TDRE flag is cleared to 0, the data is transferred from TDR to TSR, the stop bit is sent, and then serial transmission of the next frame is started.

If the TDRE flag is set to 1, the TEND flag in SSR is set to 1, the stop bit is sent, and then the “mark state” is entered in which 1 is output continuously. If the TEIE bit in SCR is set to 1 at this time, a TEI interrupt request is generated.

Figure 12.6 shows an example of the operation for transmission in asynchronous mode.

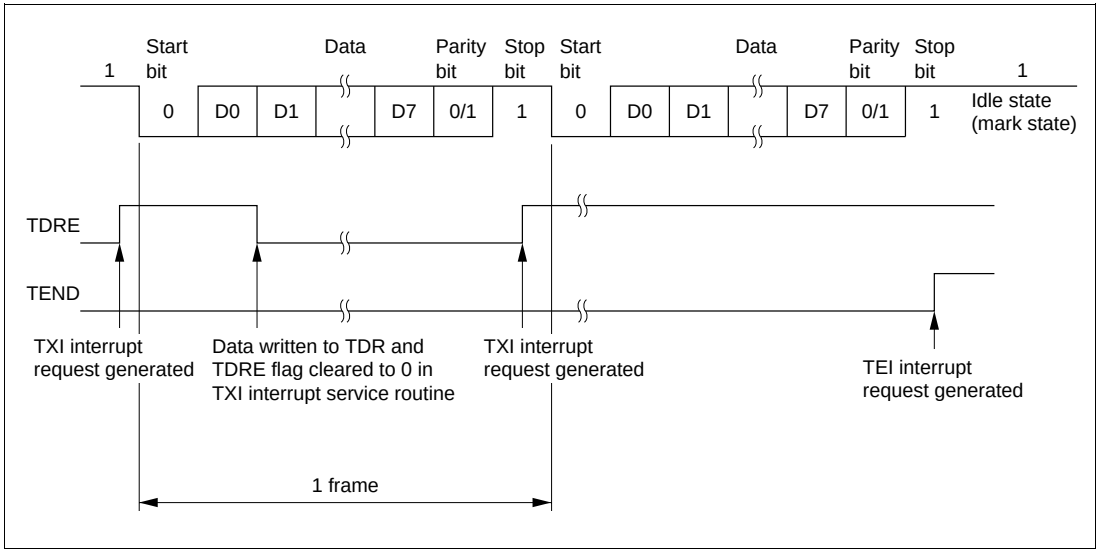
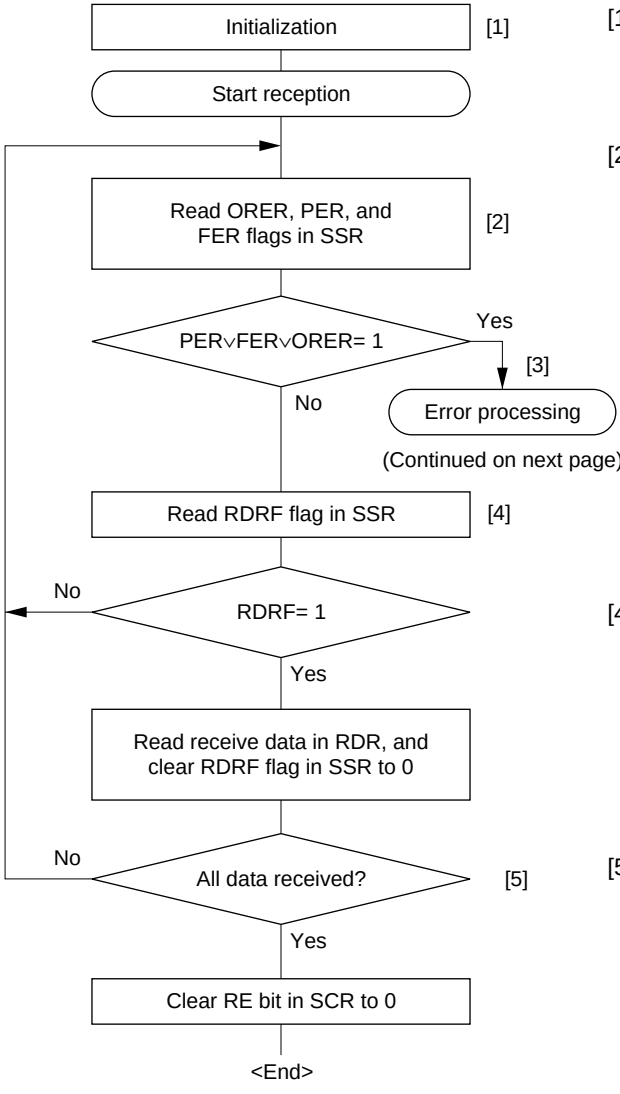



**Figure 12.6 Example of Operation in Transmission in Asynchronous Mode (Example with 8-Bit Data, Parity, One Stop Bit)**

- Serial data reception (asynchronous mode)

  Figure 12.7 shows a sample flowchart for serial reception.

  The following procedure should be used for serial data reception.




[1] SCI initialization:
The RxD pin is automatically designated as the receive data input pin.

[2] [3] Receive error processing and break detection:
If a receive error occurs, read the ORER, PER, and FER flags in SSR to identify the error. After performing the appropriate error processing, ensure that the ORER, PER, and FER flags are all cleared to 0. Reception cannot be resumed if any of these flags are set to 1. In the case of a framing error, a break can be detected by reading the value of the input port corresponding to the RxD pin.

[4] SCI status check and receive data read :
Read SSR and check that RDRF = 1, then read the receive data in RDR and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.

[5] Serial reception continuation procedure:
To continue serial reception, before the stop bit for the current frame is received, read the RDRF flag, read RDR, and clear the RDRF flag to 0. The RDRF flag is cleared automatically when the DTC is activated by an RXI interrupt and the RDR value is read.

**Figure 12.7 Sample Serial Reception Data Flowchart**

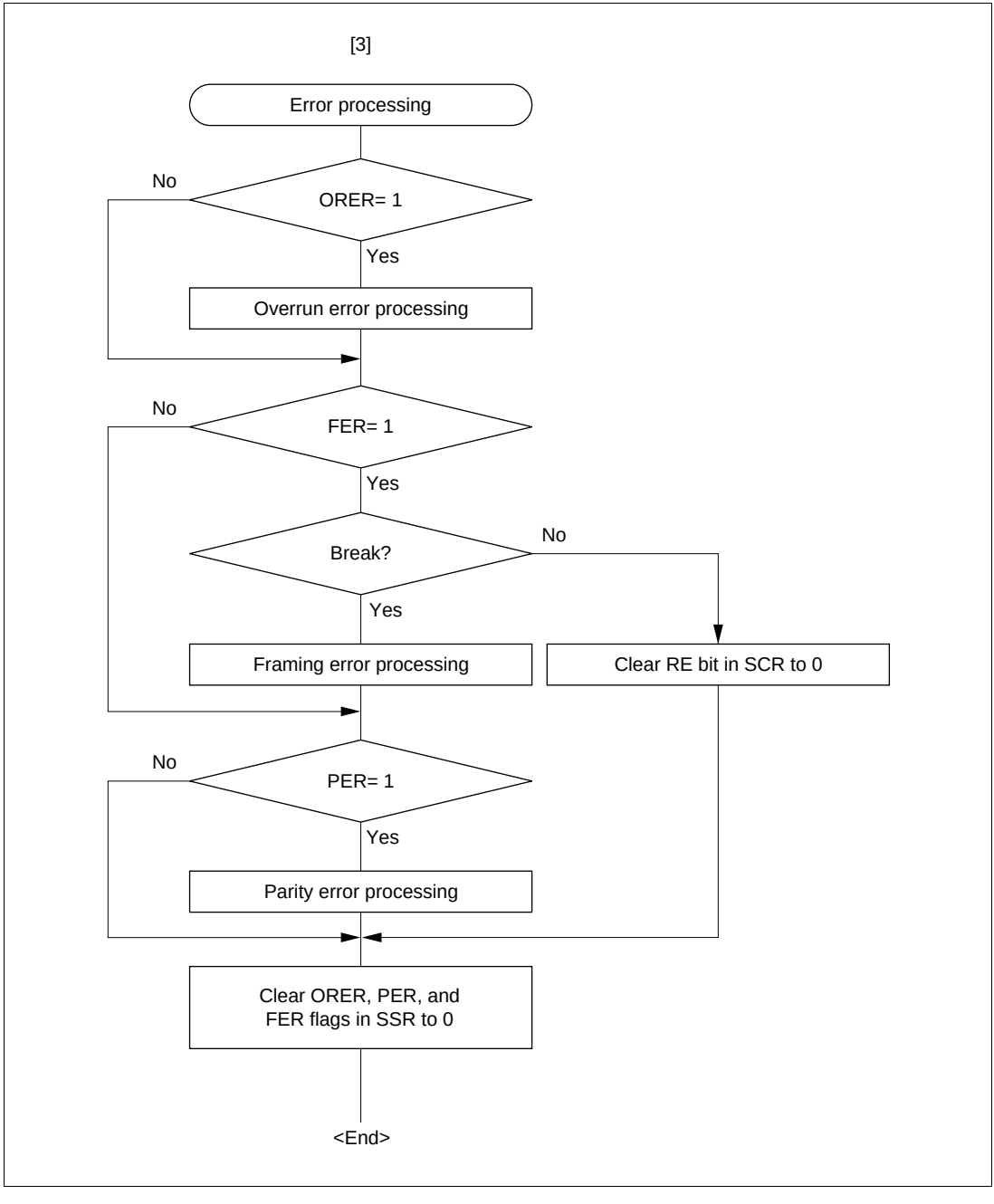



**Figure 12.7 Sample Serial Reception Data Flowchart (cont)**

In serial reception, the SCI operates as described below.

[1] The SCI monitors the transmission line, and if a 0 stop bit is detected, performs internal synchronization and starts reception.

[2] The received data is stored in RSR in LSB-to-MSB order.

[3] The parity bit and stop bit are received.

After receiving these bits, the SCI carries out the following checks.

[a] Parity check:

The SCI checks whether the number of 1 bits in the receive data agrees with the parity (even or odd) set in the O/$\overline{E}$ bit in SMR.

[b] Stop bit check:

The SCI checks whether the stop bit is 1.

If there are two stop bits, only the first is checked.

[c] Status check:

The SCI checks whether the RDRF flag is 0, indicating that the receive data can be transferred from RSR to RDR.

If all the above checks are passed, the RDRF flag is set to 1, and the receive data is stored in RDR.

If a receive error* is detected in the error check, the operation is as shown in table 12.11.

Note: * Subsequent receive operations cannot be performed when a receive error has occurred. Also note that the RDRF flag is not set to 1 in reception, and so the error flags must be cleared to 0.

[4] If the RIE bit in SCR is set to 1 when the RDRF flag changes to 1, a receive data full interrupt (RXI) request is generated.

Also, if the RIE bit in SCR is set to 1 when the ORER, PER, or FER flag changes to 1, a receive error interrupt (ERI) request is generated.

**Table 12.11 Receive Errors and Conditions for Occurrence**

| Receive Error | Abbreviation | Occurrence Condition | Data Transfer |
|---|---|---|---|
| Overrun error | ORER | When the next data reception is completed while the RDRF flag in SSR is set to 1 | Receive data is not transferred from RSR to RDR. |
| Framing error | FER | When the stop bit is 0 | Receive data is transferred from RSR to RDR. |
| Parity error | PER | When the received data differs from the parity (even or odd) set in SMR | Receive data is transferred from RSR to RDR. |

Figure 12.8 shows an example of the operation for reception in asynchronous mode.

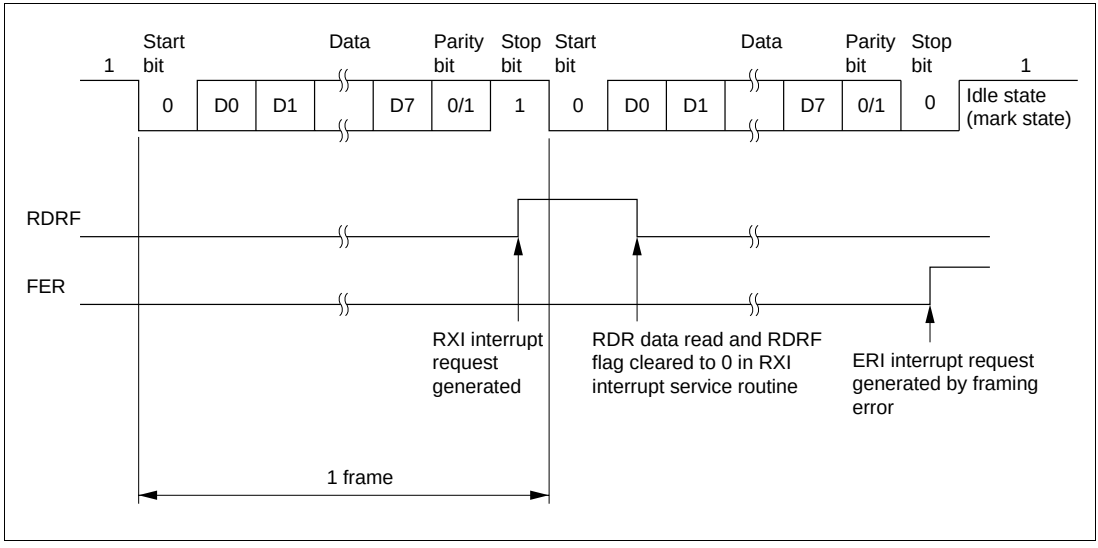



**Figure 12.8 Example of SCI Operation in Reception (Example with 8-Bit Data, Parity, One Stop Bit)**

### 12.3.3 Multiprocessor Communication Function

The multiprocessor communication function performs serial communication using the multiprocessor format, in which a multiprocessor bit is added to the transfer data, in asynchronous mode. Use of this function enables data transfer to be performed among a number of processors sharing transmission lines.

When multiprocessor communication is carried out, each receiving station is addressed by a unique ID code.

The serial communication cycle consists of two component cycles: an ID transmission cycle which specifies the receiving station , and a data transmission cycle. The multiprocessor bit is used to differentiate between the ID transmission cycle and the data transmission cycle.

The transmitting station first sends the ID of the receiving station with which it wants to perform serial communication as data with a 1 multiprocessor bit added. It then sends transmit data as data with a 0 multiprocessor bit added.

The receiving station skips the data until data with a 1 multiprocessor bit is sent.

When data with a 1 multiprocessor bit is received, the receiving station compares that data with its own ID. The station whose ID matches then receives the data sent next. Stations whose ID does not match continue to skip the data until data with a 1 multiprocessor bit is again received. In this way, data communication is carried out among a number of processors.

Figure 12.9 shows an example of inter-processor communication using the multiprocessor format.

**Data Transfer Format:** There are four data transfer formats.

When the multiprocessor format is specified, the parity bit specification is invalid.

For details, see table 12.10.

**Clock:** See the section on asynchronous mode.

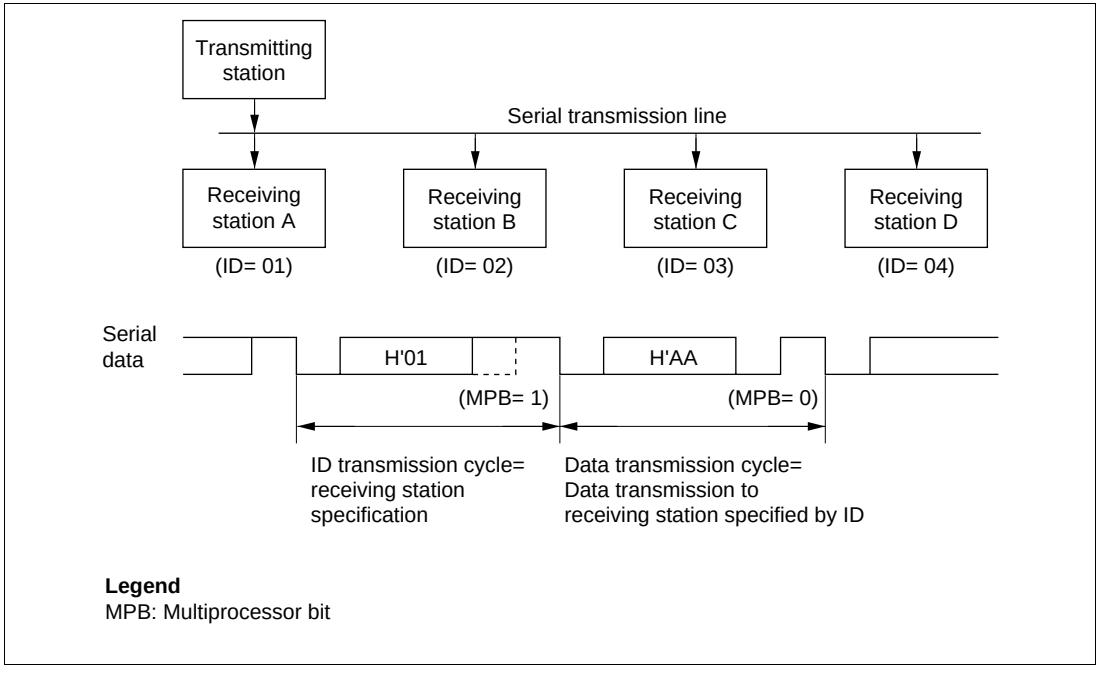


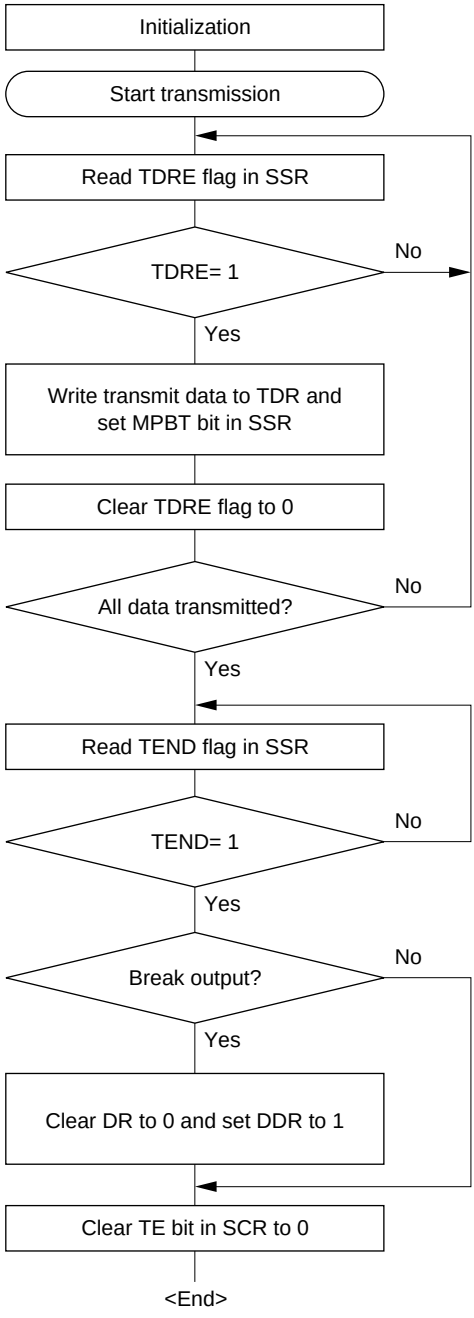

**Figure 12.9 Example of Inter-Processor Communication Using Multiprocessor Format (Transmission of Data H'AA to Receiving Station A)**

**Data Transfer Operations:**

- Multiprocessor serial data transmission

  Figure 12.10 shows a sample flowchart for multiprocessor serial data transmission.

  The following procedure should be used for multiprocessor serial data transmission.




[1] SCI initialization:
The TxD pin is automatically designated as the transmit data output pin.
After the TE bit is set to 1, a frame of 1s is output, and transmission is enabled.

[2] SCI status check and transmit data write:
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR. Set the MPBT bit in SSR to 0 or 1. Finally, clear the TDRE flag to 0.

[3] Serial transmission continuation procedure:
To continue serial transmission, be sure to read 1 from the TDRE flag to confirm that writing is possible, then write data to TDR, and then clear the TDRE flag to 0. Checking and clearing of the TDRE flag is automatic when the DTC is activated by a transmit data empty interrupt (TXI) request, and data is written to TDR.

[4] Break output at the end of serial transmission:
To output a break in serial transmission, set the port DDR to 1, clear DR to 0, then clear the TE bit in SCR to 0.

**Figure 12.10 Sample Multiprocessor Serial Transmission Flowchart**

In serial transmission, the SCI operates as described below.

[1] The SCI monitors the TDRE flag in SSR, and if is 0, recognizes that data has been written to TDR, and transfers the data from TDR to TSR.

[2] After transferring data from TDR to TSR, the SCI sets the TDRE flag to 1 and starts transmission.

If the TIE bit is set to 1 at this time, a transmit data empty interrupt (TXI) is generated.

The serial transmit data is sent from the TxD pin in the following order.

[a] Start bit:

One 0-bit is output.

[b] Transmit data:

8-bit or 7-bit data is output in LSB-first order.

[c] Multiprocessor bit

One multiprocessor bit (MPBT value) is output.

[d] Stop bit(s):

One or two 1-bits (stop bits) are output.

[e] Mark state:

1 is output continuously until the start bit that starts the next transmission is sent.

[3] The SCI checks the TDRE flag at the timing for sending the stop bit.

If the TDRE flag is cleared to 0, data is transferred from TDR to TSR, the stop bit is sent, and then serial transmission of the next frame is started.

If the TDRE flag is set to 1, the TEND flag in SSR is set to 1, the stop bit is sent, and then the mark state is entered in which 1 is output continuously. If the TEIE bit in SCR is set to 1 at this time, a transmission end interrupt (TEI) request is generated.

Figure 12.11 shows an example of SCI operation for transmission using the multiprocessor format.

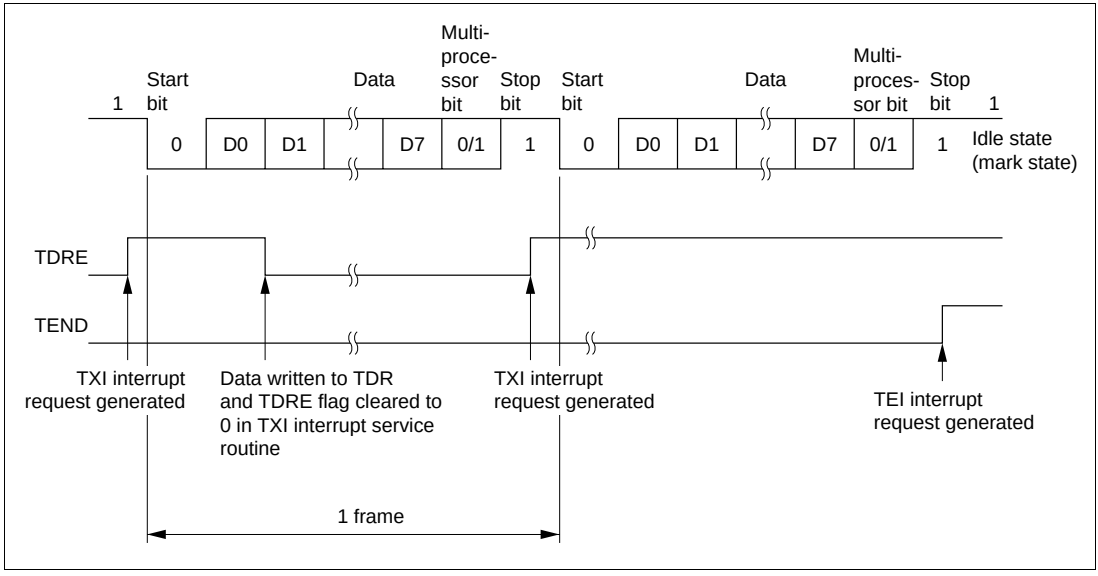
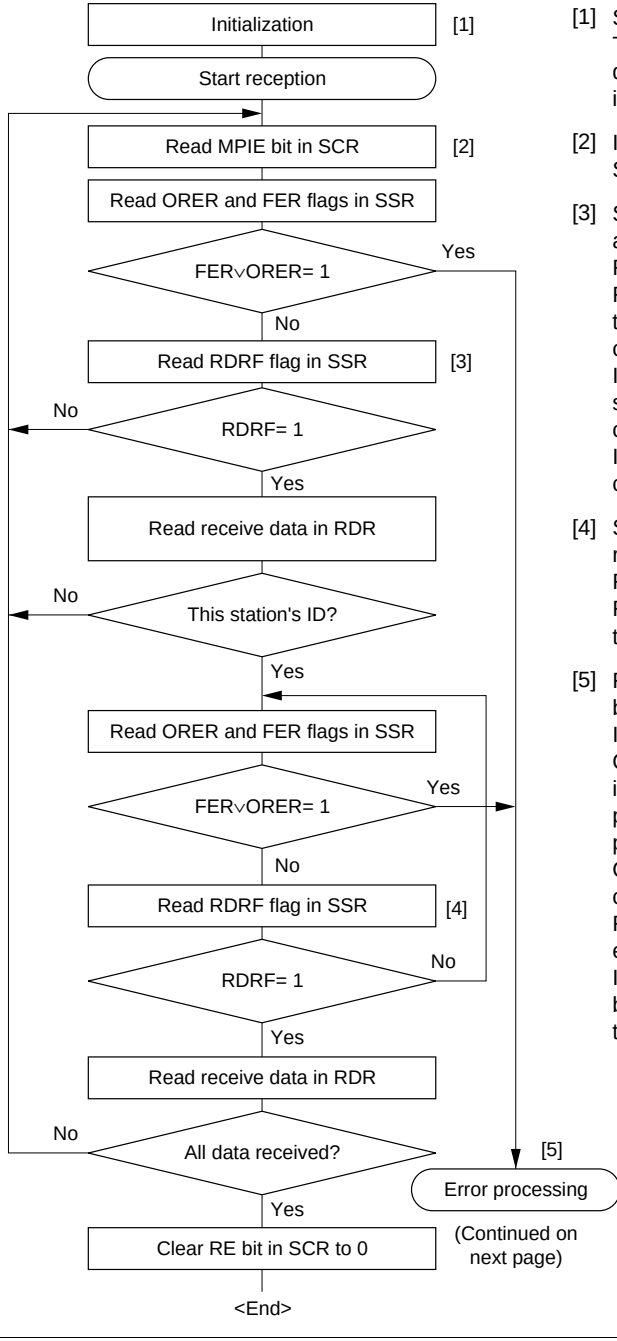



**Figure 12.11 Example of SCI Operation in Transmission (Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit)**

- Multiprocessor serial data reception

  Figure 12.12 shows a sample flowchart for multiprocessor serial reception.

  The following procedure should be used for multiprocessor serial data reception.




[1] SCI initialization:
The RxD pin is automatically designated as the receive data input pin.

[2] ID reception cycle:
Set the MPIE bit in SCR to 1.

[3] SCI status check, ID reception and comparison:
Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR and compare it with this station's ID.
If the data is not this station's ID, set the MPIE bit to 1 again, and clear the RDRF flag to 0.
If the data is this station's ID, clear the RDRF flag to 0.

[4] SCI status check and data reception:
Read SSR and check that the RDRF flag is set to 1, then read the data in RDR.

[5] Receive error processing and break detection:
If a receive error occurs, read the ORER and FER flags in SSR to identify the error. After performing the appropriate error processing, ensure that the ORER and FER flags are all cleared to 0.
Reception cannot be resumed if either of these flags is set to 1.
In the case of a framing error, a break can be detected by reading the RxD pin value.

**Figure 12.12 Sample Multiprocessor Serial Reception Flowchart**

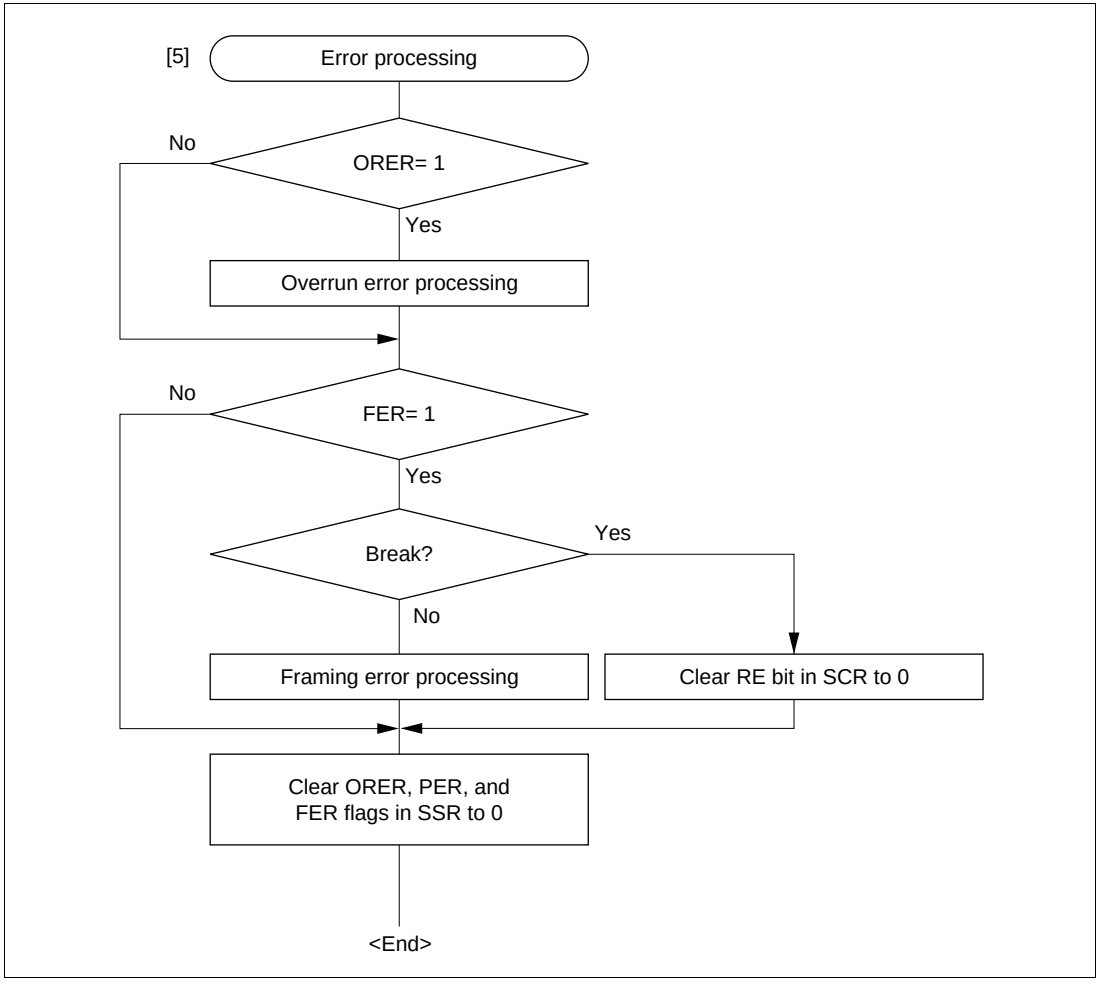



**Figure 12.12 Sample Multiprocessor Serial Reception Flowchart (cont)**

Figure 12.13 shows an example of SCI operation for multiprocessor format reception.

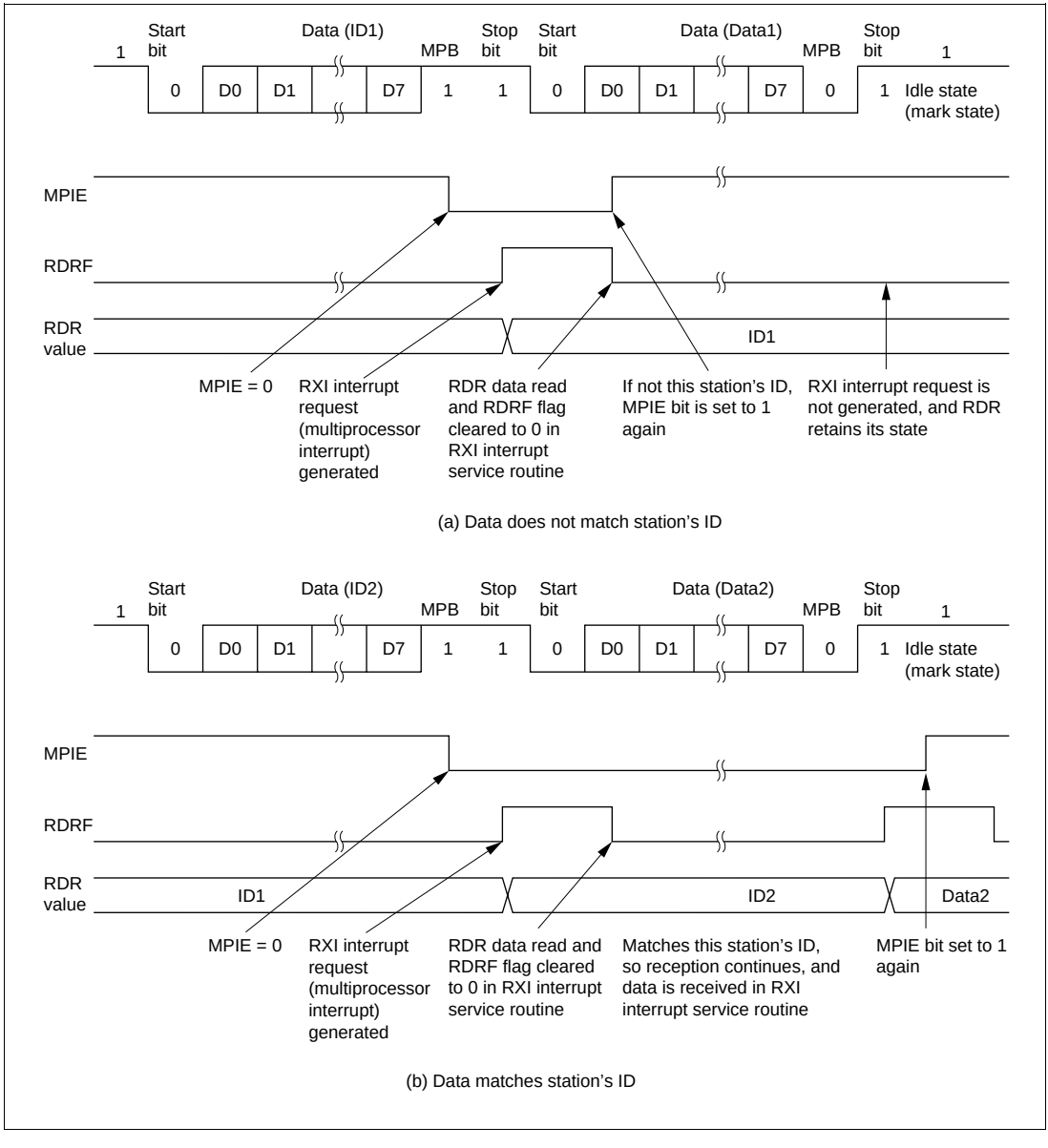



**Figure 12.13 Example of SCI Operation in Reception (Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit)**

### 12.3.4 Operation in Clocked Synchronous Mode

In clocked synchronous mode, data is transmitted or received in synchronization with clock pulses, making it suitable for high-speed serial communication.

Inside the SCI, the transmitter and receiver are independent units, enabling full-duplex communication by use of a common clock. Both the transmitter and the receiver also have a double-buffered structure, so that data can be read or written during transmission or reception, enabling continuous data transfer.

Figure 12.14 shows the general format for clocked synchronous serial communication.

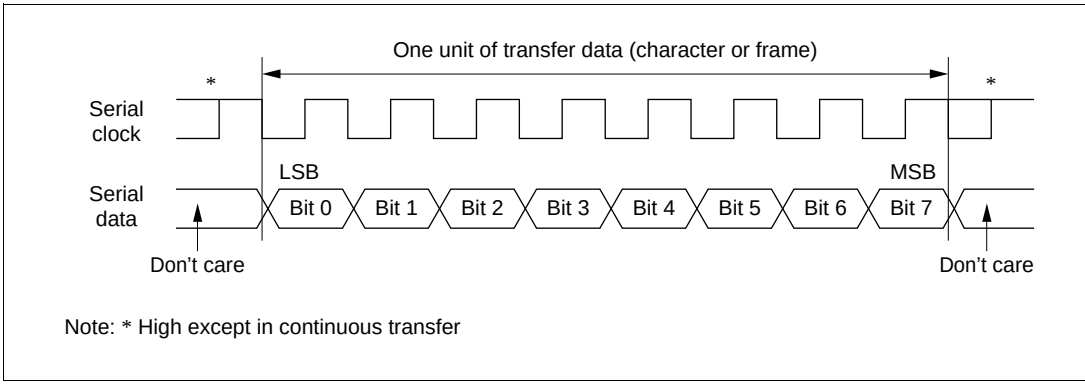



**Figure 12.14 Data Format in Synchronous Communication**

In clocked synchronous serial communication, data on the transmission line is output from one falling edge of the serial clock to the next. Data confirmation is guaranteed at the rising edge of the serial clock.

In clocked serial communication, one character consists of data output starting with the LSB and ending with the MSB. After the MSB is output, the transmission line holds the MSB state.

In clocked synchronous mode, the SCI receives data in synchronization with the rising edge of the serial clock.

**Data Transfer Format:** A fixed 8-bit data format is used.

No parity or multiprocessor bits are added.

**Clock:** Either an internal clock generated by the on-chip baud rate generator or an external serial clock input at the SCK pin can be selected, according to the setting of the C/$\overline{A}$ bit in SMR and the CKE1 and CKE0 bits in SCR. For details of SCI clock source selection, see table 12.9.

When the SCI is operated on an internal clock, the serial clock is output from the SCK pin.

Eight serial clock pulses are output in the transfer of one character, and when no transfer is performed the clock is fixed high. When only receive operations are performed, however, the serial clock is output until an overrun error occurs or the RE bit is cleared to 0. If you want to perform receive operations in units of one character, you should select an external clock as the clock source.

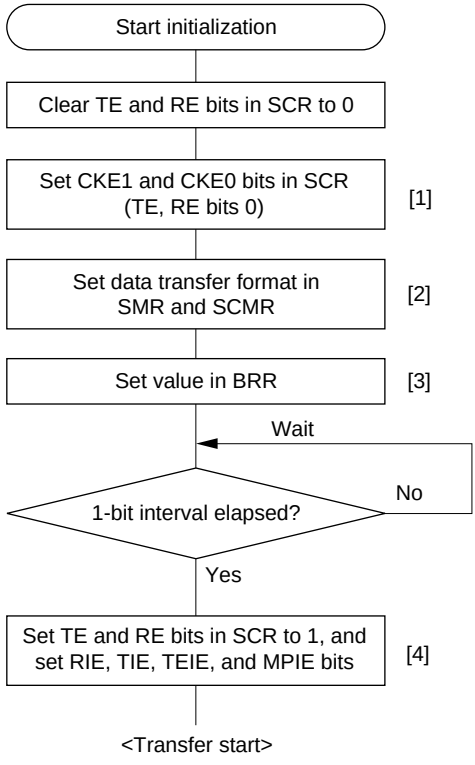
**Data Transfer Operations:**

- SCI initialization (clocked synchronous mode)

  Before transmitting and receiving data, you should first clear the TE and RE bits in SCR to 0, then initialize the SCI as described below.

  When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change using the following procedure. When the TE bit is cleared to 0, the TDRE flag is set to 1 and TSR is initialized. Note that clearing the RE bit to 0 does not change the contents of the RDRF, PER, FER, and ORER flags, or the contents of RDR.

  Figure 12.15 shows a sample SCI initialization flowchart.




[1] Set the clock selection in SCR. Be sure to clear bits RIE, TIE, TEIE, and MPIE, TE and RE, to 0.

[2] Set the data transfer format in SMR and SCMR.

[3] Write a value corresponding to the bit rate to BRR. Not necessary if an external clock is used.

[4] Wait at least one bit interval, then set the TE bit or RE bit in SCR to 1. Also set the RIE, TIE, TEIE, and MPIE bits.
Setting the TE and RE bits enables the TxD and RxD pins to be used.

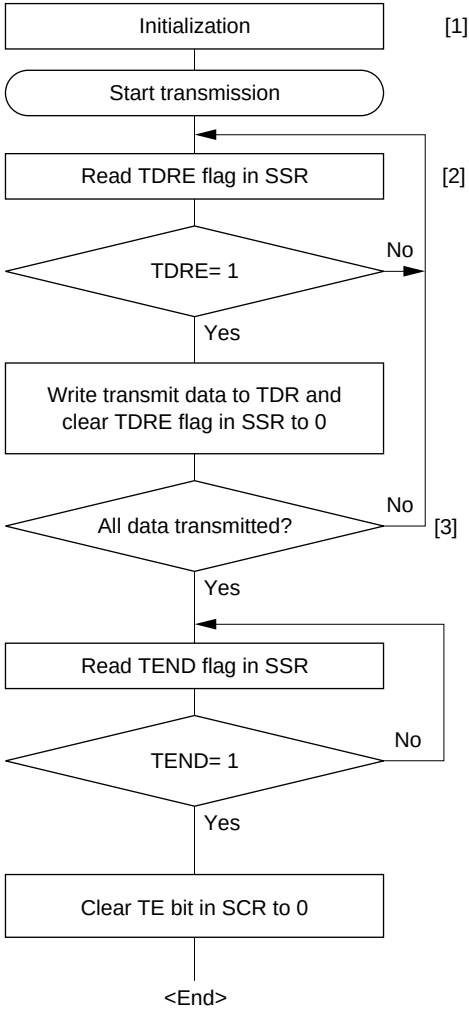
Note: * When transmitting and receiving data simultaneously, the TE and RE bits should be cleared to 0 and then set to 1 at the same time.

**Figure 12.15 Sample SCI Initialization Flowchart**

- Serial data transmission (clocked synchronous mode)

  Figure 12.16 shows a sample flowchart for serial transmission.

  The following procedure should be used for serial data transmission.




[1] SCI initialization:
The TxD pin is automatically designated as the transmit data output pin.

[2] SCI status check and transmit data write:
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR and clear the TDRE flag to 0.

[3] Serial transmission continuation procedure:
To continue serial transmission, be sure to read 1 from the TDRE flag to confirm that writing is possible, then write data to TDR, and then clear the TDRE flag to 0.
Checking and clearing of the TDRE flag is automatic when the DTC is activated by a transmit data empty interrupt (TXI) request and data is written to TDR.

**Figure 12.16 Sample Serial Transmission Flowchart**

In serial transmission, the SCI operates as described below.

[1] The SCI monitors the TDRE flag in SSR, and if is 0, recognizes that data has been written to TDR, and transfers the data from TDR to TSR.

[2] After transferring data from TDR to TSR, the SCI sets the TDRE flag to 1 and starts transmission. If the TIE bit is set to 1 at this time, a transmit data empty interrupt (TXI) is generated.

When clock output mode has been set, the SCI outputs 8 serial clock pulses. When use of an external clock has been specified, data is output synchronized with the input clock.

The serial transmit data is sent from the TxD pin starting with the LSB (bit 0) and ending with the MSB (bit 7).

[3] The SCI checks the TDRE flag at the timing for sending the MSB (bit 7).

If the TDRE flag is cleared to 0, data is transferred from TDR to TSR, and serial transmission of the next frame is started.

If the TDRE flag is set to 1, the TEND flag in SSR is set to 1, the MSB (bit 7) is sent, and the TxD pin maintains its state.

If the TEIE bit in SCR is set to 1 at this time, a TEI interrupt request is generated.

[4] After completion of serial transmission, the SCK pin is fixed.

Figure 12.17 shows an example of SCI operation in transmission.

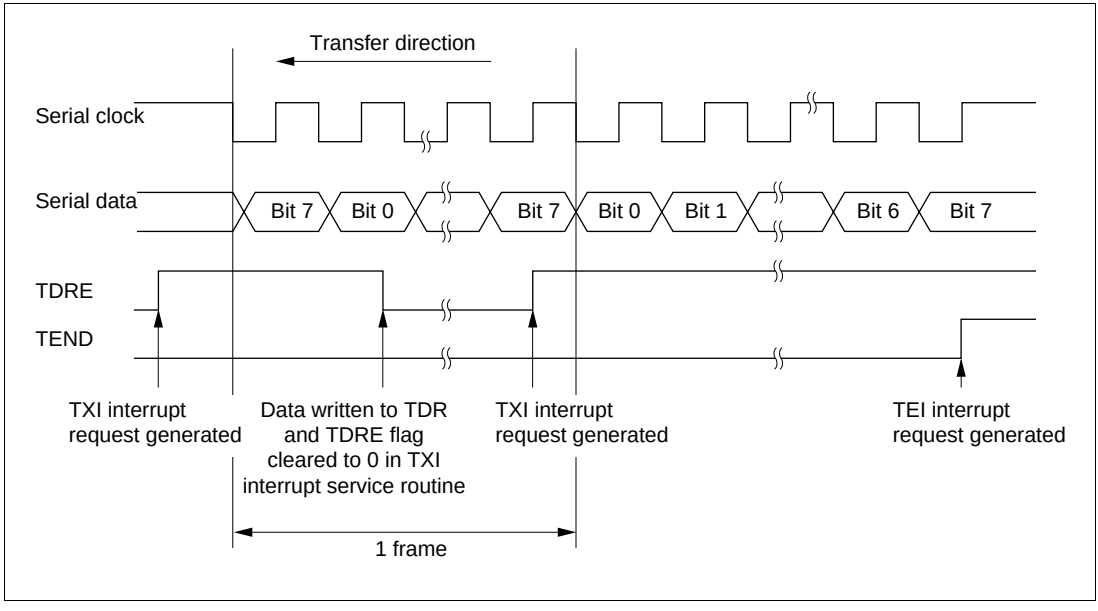



**Figure 12.17 Example of SCI Operation in Transmission**

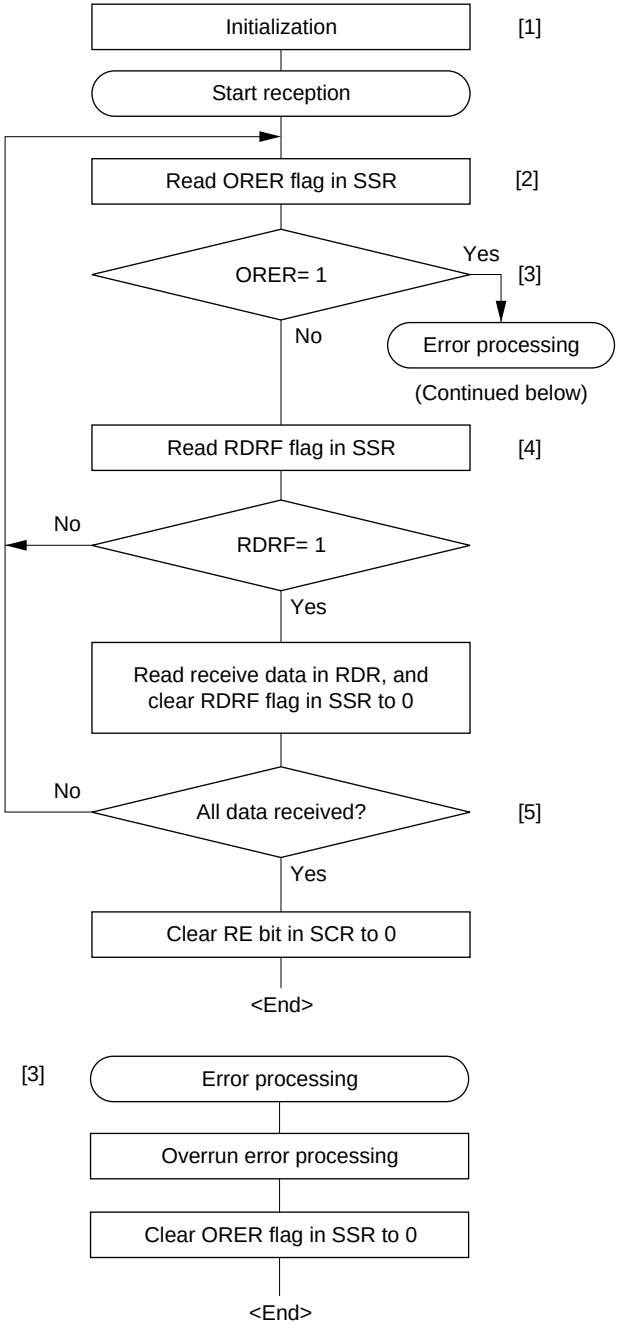
- Serial data reception (clocked synchronous mode)

Figure 12.18 shows a sample flowchart for serial reception.

The following procedure should be used for serial data reception.

When changing the operating mode from asynchronous to clocked synchronous, be sure to check that the ORER, PER, and FER flags are all cleared to 0.

The RDRF flag will not be set if the FER or PER flag is set to 1, and neither transmit nor receive operations will be possible.




[1] SCI initialization:
The RxD pin is automatically designated as the receive data input pin.

[2] [3] Receive error processing:
If a receive error occurs, read the ORER flag in SSR , and after performing the appropriate error processing, clear the ORER flag to 0. Transfer cannot be resumed if the ORER flag is set to 1.

[4] SCI status check and receive data read:
Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.

[5] Serial reception continuation procedure:
To continue serial reception, before the MSB (bit 7) of the current frame is received, finish reading the RDRF flag, reading RDR, and clearing the RDRF flag to 0. The RDRF flag is cleared automatically when the DTC is activated by a receive data full interrupt (RXI) request and the RDR value is read.

**Figure 12.18 Sample Serial Reception Flowchart**

In serial reception, the SCI operates as described below.

[1] The SCI performs internal initialization in synchronization with serial clock input or output.

[2] The received data is stored in RSR in LSB-to-MSB order.

After reception, the SCI checks whether the RDRF flag is 0 and the receive data can be transferred from RSR to RDR.

If this check is passed, the RDRF flag is set to 1, and the receive data is stored in RDR. If a receive error is detected in the error check, the operation is as shown in table 12.11.

Neither transmit nor receive operations can be performed subsequently when a receive error has been found in the error check.

[3] If the RIE bit in SCR is set to 1 when the RDRF flag changes to 1, a receive data full interrupt (RXI) request is generated.

Also, if the RIE bit in SCR is set to 1 when the ORER flag changes to 1, a receive error interrupt (ERI) request is generated.

Figure 12.19 shows an example of SCI operation in reception.

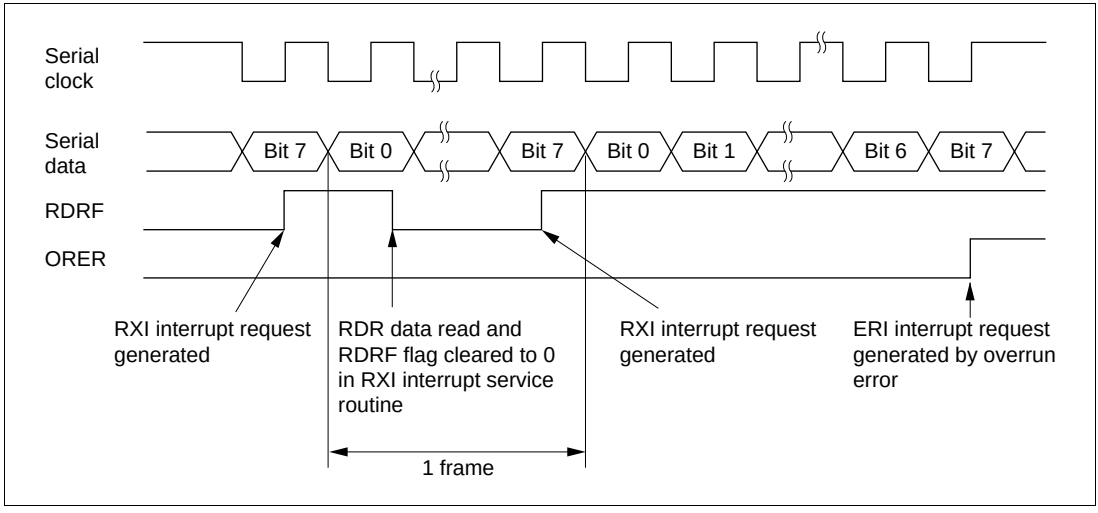
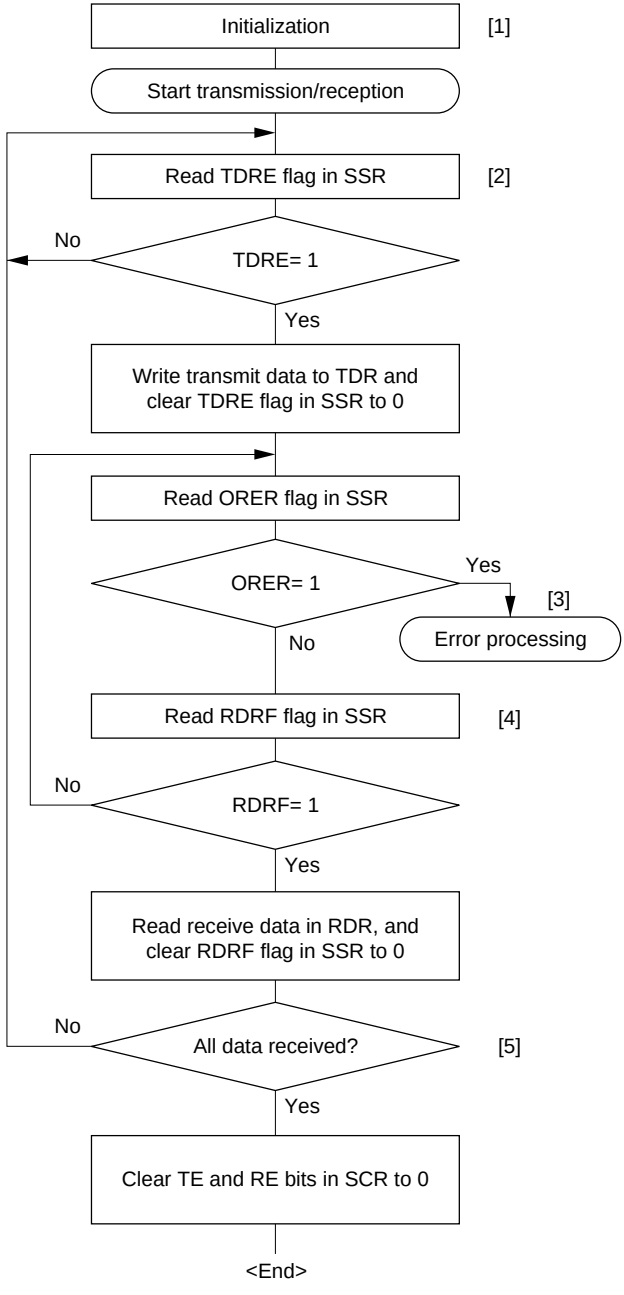



**Figure 12.19 Example of SCI Operation in Reception**

- Simultaneous serial data transmission and reception (clocked synchronous mode)

  Figure 12.20 shows a sample flowchart for simultaneous serial transmit and receive operations.

  The following procedure should be used for simultaneous serial data transmit and receive operations.




Note: When switching from transmit or receive operation to simultaneous transmit and receive operations, first clear the TE and RE bits to 0, then set both of these bits to 1.

[1] SCI initialization:
The TxD pin is designated as the transmit data output pin, and the RxD pin is designated as the receive data input pin, enabling simultaneous transmit and receive operations.

[2] SCI status check and transmit data write:
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR and clear the TDRE flag to 0.
Transition of the TDRE flag from 0 to 1 can also be identified by a TXI interrupt.

[3] Receive error processing:
If a receive error occurs, read the ORER flag in SSR , and after performing the appropriate error processing, clear the ORER flag to 0. Transmission/reception cannot be resumed if the ORER flag is set to 1.

[4] SCI status check and receive data read:
Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.

[5] Serial transmission/reception continuation procedure:
To continue serial transmission/reception, before the MSB (bit 7) of the current frame is received, finish reading the RDRF flag, reading RDR, and clearing the RDRF flag to 0. Also, before the MSB (bit 7) of the current frame is transmitted, read 1 from the TDRE flag to confirm that writing is possible. Then write data to TDR and clear the TDRE flag to 0.
Checking and clearing of the TDRE flag is automatic when the DTC is activated by a transmit data empty interrupt (TXI) request and data is written to TDR. Also, the RDRF flag is cleared automatically when the DTC is activated by a receive data full interrupt (RXI) request and the RDR value is read.

**Figure 12.20 Sample Flowchart of Simultaneous Serial Transmit and Receive Operations**

## 12.4 SCI Interrupts

The SCI has four interrupt sources: the transmit-end interrupt (TEI) request, receive-error interrupt (ERI) request, receive-data-full interrupt (RXI) request, and transmit-data-empty interrupt (TXI) request. Table 12.12 shows the interrupt sources and their relative priorities. Individual interrupt sources can be enabled or disabled with the TIE, RIE, and TEIE bits in the SCR. Each kind of interrupt request is sent to the interrupt controller independently.

When the TDRE flag in SSR is set to 1, a TXI interrupt request is generated. When the TEND flag in SSR is set to 1, a TEI interrupt request is generated. A TXI interrupt can activate the DTC to perform data transfer. The TDRE flag is cleared to 0 automatically when data transfer is performed by the DTC. The DTC cannot be activated by a TEI interrupt request.

When the RDRF flag in SSR is set to 1, an RXI interrupt request is generated. When the ORER, PER, or FER flag in SSR is set to 1, an ERI interrupt request is generated. An RXI interrupt can activate the DTC to perform data transfer. The RDRF flag is cleared to 0 automatically when data transfer is performed by the DTC. The DTC cannot be activated by an ERI interrupt request.

**Table 12.12 SCI Interrupt Sources**

| Channel | Interrupt Source | Description | DTC Activation | Priority* |
|---|---|---|---|---|
| 0 | ERI | Interrupt due to receive error (ORER, FER, or PER) | Not possible | High |
| | RXI | Interrupt due to receive data full state (RDRF) | Possible | |
| | TXI | Interrupt due to transmit data empty state (TDRE) | Possible | |
| | TEI | Interrupt due to transmission end (TEND) | Not possible | |
| 1 | ERI | Interrupt due to receive error (ORER, FER, or PER) | Not possible | |
| | RXI | Interrupt due to receive data full state (RDRF) | Possible | |
| | TXI | Interrupt due to transmit data empty state (TDRE) | Possible | |
| | TEI | Interrupt due to transmission end (TEND) | Not possible | Low |

Note: * This table shows the initial state immediately after a reset. Relative priorities among channels can be changed by means of ICR and IPR.

A TEI interrupt is requested when the TEND flag is set to 1 while the TEIE bit is set to 1. The TEND flag is cleared at the same time as the TDRE flag. Consequently, if a TEI interrupt and a TXI interrupt are requested simultaneously, the TXI interrupt may be accepted first, with the result that the TDRE and TEND flags are cleared. Note that the TEI interrupt will not be accepted in this case.

## 12.5 Usage Notes

The following points should be noted when using the SCI.

**Relation between Writes to TDR and the TDRE Flag**

The TDRE flag in SSR is a status flag that indicates that transmit data has been transferred from TDR to TSR. When the SCI transfers data from TDR to TSR, the TDRE flag is set to 1.

Data can be written to TDR regardless of the state of the TDRE flag. However, if new data is written to TDR when the TDRE flag is cleared to 0, the data stored in TDR will be lost since it has not yet been transferred to TSR. It is therefore essential to check that the TDRE flag is set to 1 before writing transmit data to TDR.

**Operation when Multiple Receive Errors Occur Simultaneously**

If a number of receive errors occur at the same time, the state of the status flags in SSR is as shown in table 12.13. If there is an overrun error, data is not transferred from RSR to RDR, and the receive data is lost.

**Table 12.13 State of SSR Status Flags and Transfer of Receive Data**

| SSR Status Flags | | | | Receive Data Transfer | |
|---|---|---|---|---|---|
| RDRF | ORER | FER | PER | RSR to RDR | Receive Error Status |
| 1 | 1 | 0 | 0 | X | Overrun error |
| 0 | 0 | 1 | 0 | ○ | Framing error |
| 0 | 0 | 0 | 1 | ○ | Parity error |
| 1 | 1 | 1 | 0 | X | Overrun error + framing error |
| 1 | 1 | 0 | 1 | X | Overrun error + parity error |
| 0 | 0 | 1 | 1 | ○ | Framing error + parity error |
| 1 | 1 | 1 | 1 | X | Overrun error + framing error + parity error |

Notes: ○: Receive data is transferred from RSR to RDR.
X: Receive data is not transferred from RSR to RDR.

**Break Detection and Processing (Asynchronous Mode Only):** When framing error (FER) detection is performed, a break can be detected by reading the RxD pin value directly. In a break, the input from the RxD pin becomes all 0s, and so the FER flag is set, and the parity error flag (PER) may also be set.

Note that, since the SCI continues the receive operation after receiving a break, even if the FER flag is cleared to 0, it will be set to 1 again.

**Sending a Break (Asynchronous Mode Only):** The TxD pin has a dual function as an I/O port whose direction (input or output) is determined by DR and DDR. This can be used to send a break.

Between serial transmission initialization and setting of the TE bit to 1, the mark state is replaced by the value of DR (the pin does not function as the TxD pin until the TE bit is set to 1). Consequently, DDR and DR for the port corresponding to the TxD pin are first set to 1.

To send a break during serial transmission, first clear DR to 0, then clear the TE bit to 0.

When the TE bit is cleared to 0, the transmitter is initialized regardless of the current transmission state, the TxD pin becomes an I/O port, and 0 is output from the TxD pin.

**Receive Error Flags and Transmit Operations (Clocked Synchronous Mode Only):** Transmission cannot be started when a receive error flag (ORER, PER, or FER) is set to 1, even if the TDRE flag is cleared to 0. Be sure to clear the receive error flags to 0 before starting transmission.

Note also that receive error flags cannot be cleared to 0 even if the RE bit is cleared to 0.

**Receive Data Sampling Timing and Reception Margin in Asynchronous Mode:** In asynchronous mode, the SCI operates on a basic clock with a frequency of 16 times the transfer rate.

In reception, the SCI samples the falling edge of the start bit using the basic clock, and performs internal synchronization. Receive data is latched internally at the rising edge of the 8th pulse of the basic clock. This is illustrated in figure 12.21.

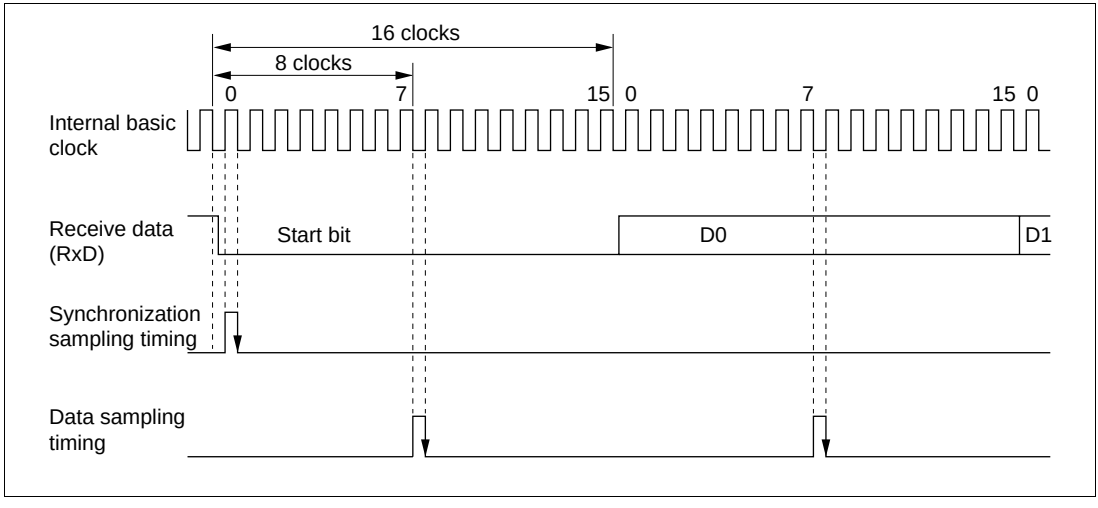



**Figure 12.21 Receive Data Sampling Timing in Asynchronous Mode**

Thus the reception margin in asynchronous mode is given by formula (1) below.

$$M = \left| (0.5 - \frac{1}{2N}) - (L - 0.5)\,F - \frac{|D - 0.5|}{N}\,(1 + F) \right| \times 100\% \quad \ldots\ldots\ldots \text{Formula (1)}$$

Where M : Reception margin (%)
N : Ratio of bit rate to clock (N = 16)
D : Clock duty (D = 0 to 1.0)
L : Frame length (L = 9 to 12)
F : Absolute value of clock rate deviation

Assuming values of F = 0 and D = 0.5 in formula (1), a reception margin of 46.875% is given by formula (2) below.

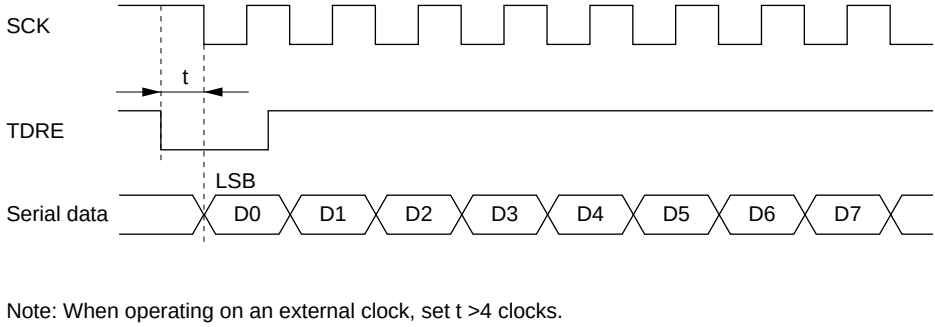
When D = 0.5 and F = 0,

$$M = (0.5 - \frac{1}{2 \times 16}) \times 100\%$$
$$= 46.875\% \quad \ldots\ldots\ldots \text{Formula (2)}$$

However, this is only the computed value, and a margin of 20% to 30% should be allowed in system design.

**Restrictions on Use of DTC**

- When an external clock source is used as the serial clock, the transmit clock should not be input until at least 5 ø clock cycles after TDR is updated by the DTC. Misoperation may occur if the transmit clock is input within 4 ø clocks after TDR is updated. (Figure 12.22)
- When RDR is read by the DTC, be sure to set the activation source to the relevant SCI reception end interrupt (RXI).




Note: When operating on an external clock, set t >4 clocks.

**Figure 12.22 Example of Clocked Synchronous Transmission by DTC**

**Interrupts and Module Stop Mode:** If module stop mode is entered when an interrupt has been requested, it will not be possible to clear the CPU interrupt source or DTC activation source. Interrupts should therefore be disabled before entering module stop mode.

# Section 13 Smart Card Interface

## 13.1 Overview

SCI supports an IC card (Smart Card) interface conforming to ISO/IEC 7816-3 (Identification Card) as a serial communication interface extension function.

Switching between the normal serial communication interface and the Smart Card interface is carried out by means of a register setting.

### 13.1.1 Features

Features of the Smart Card interface supported by the H8S/2345 Series are as follows.

- Asynchronous mode
  - Data length: 8 bits
  - Parity bit generation and checking
  - Transmission of error signal (parity error) in receive mode
  - Error signal detection and automatic data retransmission in transmit mode
  - Direct convention and inverse convention both supported
- On-chip baud rate generator allows any bit rate to be selected
- Three interrupt sources
  - Three interrupt sources (transmit data empty, receive data full, and transmit/receive error) that can issue requests independently
  - The transmit data empty interrupt and receive data full interrupt can activate the data transfer controller (DTC) to execute data transfer

### 13.1.2 Block Diagram

Figure 13.1 shows a block diagram of the Smart Card interface.

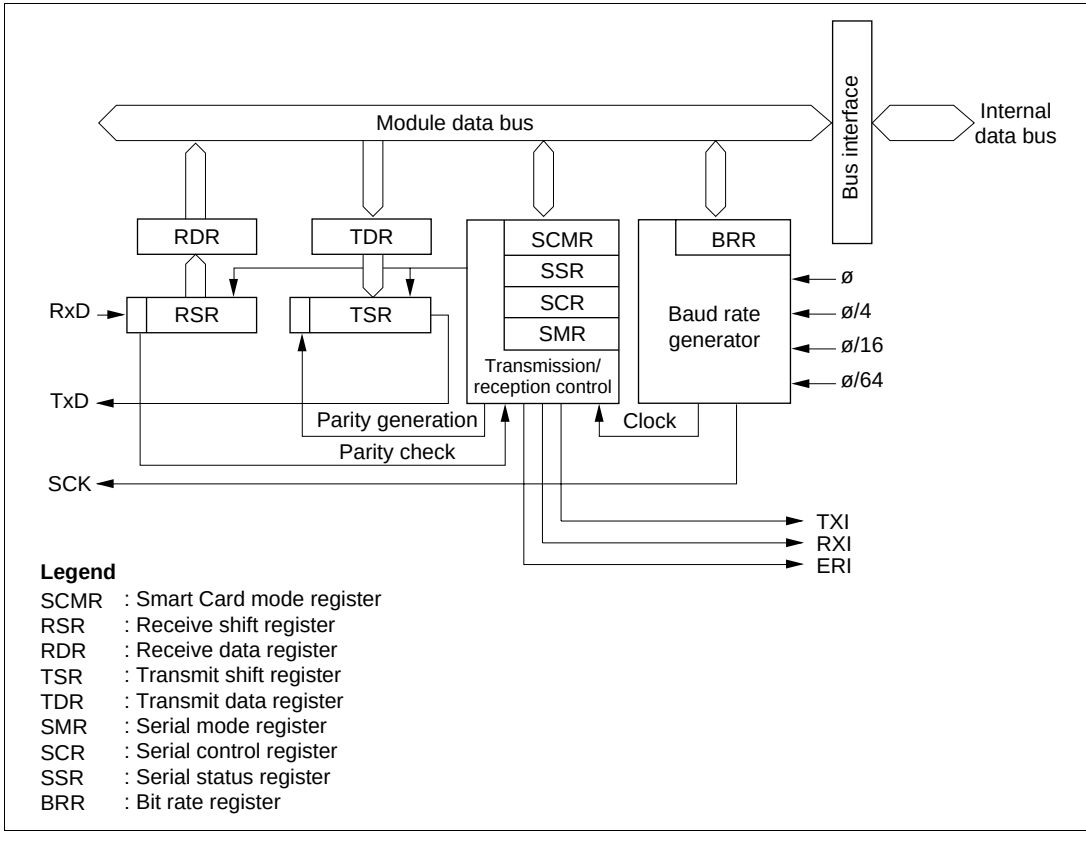



**Figure 13.1 Block Diagram of Smart Card Interface**

### 13.1.3 Pin Configuration

Table 13.1 shows the Smart Card interface pin configuration.

**Table 13.1 Smart Card Interface Pins**

| Channel | Pin Name | Symbol | I/O | Function |
|---|---|---|---|---|
| 0 | Serial clock pin 0 | SCK0 | I/O | SCI0 clock input/output |
| | Receive data pin 0 | RxD0 | Input | SCI0 receive data input |
| | Transmit data pin 0 | TxD0 | Output | SCI0 transmit data output |
| 1 | Serial clock pin 1 | SCK1 | I/O | SCI1 clock input/output |
| | Receive data pin 1 | RxD1 | Input | SCI1 receive data input |
| | Transmit data pin 1 | TxD1 | Output | SCI1 transmit data output |

### 13.1.4 Register Configuration

Table 13.2 shows the registers used by the Smart Card interface. Details of SMR, BRR, SCR, TDR, RDR, and MSTPCR are the same as for the normal SCI function: see the register descriptions in section 12, Serial Communication Interface.

**Table 13.2 Smart Card Interface Registers**

| Channel | Name | Abbreviation | R/W | Initial Value | Address*[1] |
|---|---|---|---|---|---|
| 0 | Serial mode register 0 | SMR0 | R/W | H'00 | H'FF78 |
| | Bit rate register 0 | BRR0 | R/W | H'FF | H'FF79 |
| | Serial control register 0 | SCR0 | R/W | H'00 | H'FF7A |
| | Transmit data register 0 | TDR0 | R/W | H'FF | H'FF7B |
| | Serial status register 0 | SSR0 | R/(W)*[2] | H'84 | H'FF7C |
| | Receive data register 0 | RDR0 | R | H'00 | H'FF7D |
| | Smart card mode register 0 | SCMR0 | R/W | H'F2 | H'FF7E |
| 1 | Serial mode register 1 | SMR1 | R/W | H'00 | H'FF80 |
| | Bit rate register 1 | BRR1 | R/W | H'FF | H'FF81 |
| | Serial control register 1 | SCR1 | R/W | H'00 | H'FF82 |
| | Transmit data register 1 | TDR1 | R/W | H'FF | H'FF83 |
| | Serial status register 1 | SSR1 | R/(W)*[2] | H'84 | H'FF84 |
| | Receive data register 1 | RDR1 | R | H'00 | H'FF85 |
| | Smart card mode register 1 | SCMR1 | R/W | H'F2 | H'FF86 |
| All | Module stop control register | MSTPCR | R/W | H'3FFF | H'FF3C |

Notes: 1. Lower 16 bits of the address.
2. Can only be written with 0 for flag clearing.

## 13.2 Register Descriptions

Registers added with the Smart Card interface and bits for which the function changes are described here.

### 13.2.1 Smart Card Mode Register (SCMR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | SDIR | SINV | — | SMIF |
| Initial value | : | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| R/W | : | — | — | — | — | R/W | R/W | — | R/W |

SCMR is an 8-bit readable/writable register that selects the Smart Card interface function.

SCMR is initialized to H'F2 by a reset, and in standby mode or module stop mode.

**Bits 7 to 4—Reserved:** Read-only bits, always read as 1.

**Bit 3—Smart Card Data Transfer Direction (SDIR):** Selects the serial/parallel conversion format.

**Bit 3**

| SDIR | Description | |
|---|---|---|
| 0 | TDR contents are transmitted LSB-first | (Initial value) |
| | Receive data is stored in RDR LSB-first | |
| 1 | TDR contents are transmitted MSB-first | |
| | Receive data is stored in RDR MSB-first | |

**Bit 2—Smart Card Data Invert (SINV):** Specifies inversion of the data logic level. This function is used together with the SDIR bit for communication with an inverse convention card. The SINV bit does not affect the logic level of the parity bit. For parity-related setting procedures, see section 13.3.4, Register Settings.

**Bit 2**

| SINV | Description | |
|---|---|---|
| 0 | TDR contents are transmitted as they are | (Initial value) |
| | Receive data is stored as it is in RDR | |
| 1 | TDR contents are inverted before being transmitted | |
| | Receive data is stored in inverted form in RDR | |

**Bit 1—Reserved:** Read-only bit, always read as 1.

**Bit 0—Smart Card Interface Mode Select (SMIF):** Enables or disables the Smart Card interface function.

| Bit 0 | | |
|---|---|---|
| **SMIF** | **Description** | |
| 0 | Smart Card interface function is disabled | (Initial value) |
| 1 | Smart Card interface function is enabled | |

### 13.2.2 Serial Status Register (SSR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | TDRE | RDRF | ORER | ERS | PER | TEND | MPB | MPBT |
| Initial value | : | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| R/W | : | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R | R | R/W |

Note: * Only 0 can be written to bits 7 to 3, to clear these flags.

Bit 4 of SSR has a different function in Smart Card interface mode. Coupled with this, the setting conditions for bit 2, TEND, are also different.

**Bits 7 to 5—**Operate in the same way as for the normal SCI. For details, see section 12.2.7, Serial Status Register (SSR).

**Bit 4—Error Signal Status (ERS):** In Smart Card interface mode, bit 4 indicates the status of the error signal sent back from the receiving end in transmission. Framing errors are not detected in Smart Card interface mode.

| Bit 4 | |
|---|---|
| **ERS** | **Description** |
| 0 | Indicates that data was received normally and no error signal was sent<br>[Clearing condition] (Initial value)<br>• Upon reset, and in standby mode or module stop mode<br>• When 0 is written to ERS after reading ERS = 1 |
| 1 | Indicates that an error signal was sent from the receiving side showing that a parity error was detected<br>[Setting condition]<br>When the low level of the error signal is sampled |

Note: Clearing the TE bit in SCR to 0 does not affect the ERS flag, which retains its previous state.

**Bits 3 to 0—**Operate in the same way as for the normal SCI. For details, see section 12.2.7, Serial Status Register (SSR).

However, the setting conditions for the TEND bit, are as shown below.

| Bit 2 | |
|---|---|
| **TEND** | **Description** |
| 0 | Indicates data transmission in progress<br>[Clearing conditions] (Initial value)<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DTC is activated by a TXI interrupt and writes data to TDR |
| 1 | Indicates that data transmission is finished<br>[Setting conditions]<br>• Upon reset, and in standby mode or module stop mode<br>• When the TE bit in SCR is 0 and the ERS bit is also 0<br>• When TDRE = 1 and ERS = 0 (normal transmission) 2.5 etu after a 1-byte serial character is transmitted when GM = 0<br>• When TDRE = 1 and ERS = 0 (normal transmission) 1.0 etu after a 1-byte serial character is transmitted when GM = 1. |

Note: etu: Elementary Time Unit (time for transfer of 1 bit)

### 13.2.3 Serial Mode Register (SMR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | GM | CHR | PE | O/$\overline{E}$ | STOP | MP | CKS1 | CKS0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Set value* | : | GM | 0 | 1 | O/$\overline{E}$ | 1 | 0 | CKS1 | CKS0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note: * When the smart card interface is used, be sure to make the 0 or 1 setting shown for bits 6, 5, 3, and 2.

Bit 7 of SMR has a different function in smart card interface mode.

**Bit 7—GSM Mode (GM):** Sets the smart card interface function to GSM mode.

This bit is cleared to 0 when the normal smart card interface is used. In GSM mode, this bit is set to 1, the timing of setting of the TEND flag that indicates transmission completion is advanced and clock output control mode addition is performed. The contents of the clock output control mode addition are specified by bits 1 and 0 of the serial control register (SCR).

**Bit 7**

| GM | Description |
|---|---|
| 0 | Normal smart card interface mode operation (Initial value)<br>• TEND flag generation 12.5 etu after beginning of start bit<br>• Clock output ON/OFF control only |
| 1 | GSM mode smart card interface mode operation<br>• TEND flag generation 11.0 etu after beginning of start bit<br>• High/low fixing control possible in addition to clock output ON/OFF control (set by SCR) |

Note: etu: Elementary time unit (time for transfer of 1 bit)

Bits 6 to 0—Operate in the same way as for the normal SCI.

For details, see section 12.2.5, Serial Mode Register (SMR).

### 13.2.4 Serial Control Register (SCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Bits 1 and 0 of SCR have a different function in smart card interface mode.

Bits 7 to 2—Operate in the same way as for the normal SCI.

For details, see section 12.2.6, Serial Control Register (SCR).

**Bits 1 and 0—Clock Enable (CKE1, CKE0):** Selects the clock source, and enables or disables clock output from the SCK pin.

In smart card interface mode, it is possible to switch between enabling and disabling of the normal clock output, and specify a fixed high level or fixed low level for the clock output.

| SCMR | SMR | SCR Setting | | |
|---|---|---|---|---|
| **SMIF** | **C/$\overline{A}$, GM** | **CKE1** | **CKE0** | **SCK Pin Function Description** |
| 0 | | | | Refer to SCI designation |
| 1 | 0 | 0 | 0 | The pin functions as an I/O port |
| | | | 1 | The pin outputs the clock as the SCK output pin |
| | 1 | | 0 | The pin outputs fixed low level as the SCK output pin |
| | | | 1 | The pin outputs the clock as the SCK output pin |
| | | 1 | 0 | The pin outputs fixed high level as the SCK output pin |
| | | | 1 | The pin outputs the clock as the SCK output pin |

## 13.3 Operation

### 13.3.1 Overview

The main functions of the Smart Card interface are as follows.

- One frame consists of 8-bit data plus a parity bit.
- In transmission, a guard time of at least 2 etu (Elementary Time Unit: the time for transfer of one bit) is left between the end of the parity bit and the start of the next frame.
- If a parity error is detected during reception, a low error signal level is output for one etu period, 10.5 etu after the start bit.
- If the error signal is sampled during transmission, the same data is transmitted automatically after the elapse of 2 etu or longer.
- Only start-stop asynchronous communication is supported; there is no clocked synchronous communication function.

### 13.3.2 Pin Connections

Figure 13.2 shows a schematic diagram of Smart Card interface related pin connections.

In communication with an IC card, since both transmission and reception are carried out on a single data transmission line, the TxD pin and RxD pin should be connected with the LSI pin. The data transmission line should be pulled up to the $V_{CC}$ power supply with a resistor.

When the clock generated on the Smart Card interface is used by an IC card, the SCK pin output is input to the CLK pin of the IC card. No connection is needed if the IC card uses an internal clock.

LSI port output is used as the reset signal.

Other pins must normally be connected to the power supply or ground.

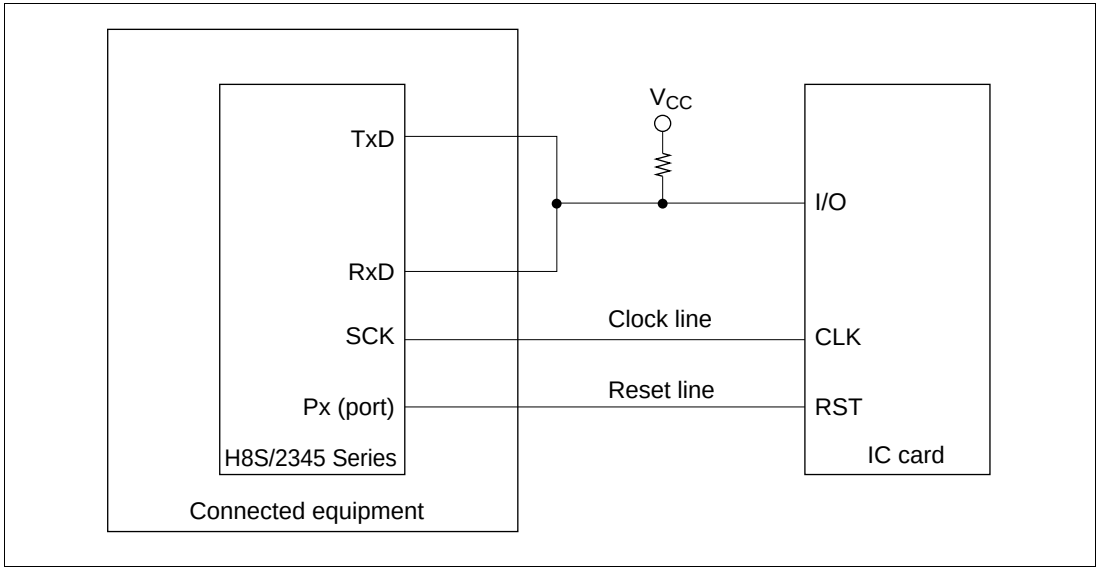



**Figure 13.2 Schematic Diagram of Smart Card Interface Pin Connections**

Note: If an IC card is not connected, and the TE and RE bits are both set to 1, closed transmission/reception is possible, enabling self-diagnosis to be carried out.

### 13.3.3 Data Format

Figure 13.3 shows the Smart Card interface data format. In reception in this mode, a parity check is carried out on each frame, and if an error is detected an error signal is sent back to the transmitting end, and retransmission of the data is requested. If an error signal is sampled during transmission, the same data is retransmitted.

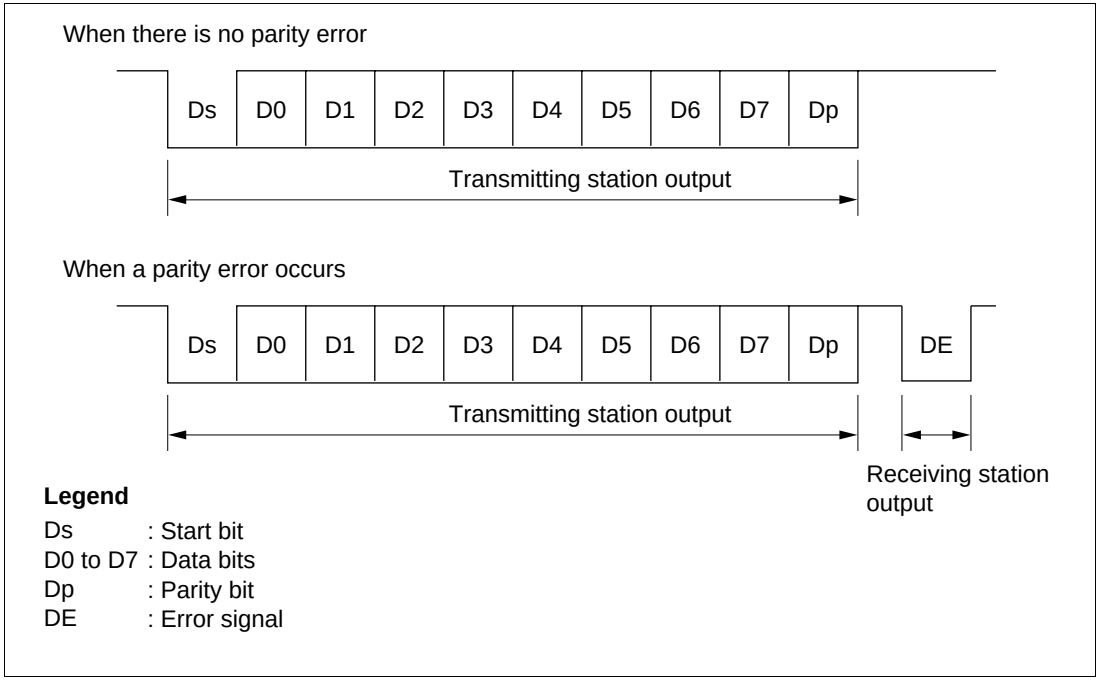



**Figure 13.3 Smart Card Interface Data Format**

The operation sequence is as follows.

[1] When the data line is not in use it is in the high-impedance state, and is fixed high with a pull-up resistor.

[2] The transmitting station starts transfer of one frame of data. The data frame starts with a start bit (Ds, low-level), followed by 8 data bits (D0 to D7) and a parity bit (Dp).

[3] With the Smart Card interface, the data line then returns to the high-impedance state. The data line is pulled high with a pull-up resistor.

[4] The receiving station carries out a parity check.

If there is no parity error and the data is received normally, the receiving station waits for reception of the next data.

If a parity error occurs, however, the receiving station outputs an error signal (DE, low-level) to request retransmission of the data. After outputting the error signal for the prescribed length of time, the receiving station places the signal line in the high-impedance state again. The signal line is pulled high again by a pull-up resistor.

[5] If the transmitting station does not receive an error signal, it proceeds to transmit the next data frame.

If it does receive an error signal, however, it returns to step [2] and retransmits the erroneous data.

### 13.3.4 Register Settings

Table 13.3 shows a bit map of the registers used by the smart card interface.

Bits indicated as 0 or 1 must be set to the value shown. The setting of other bits is described below.

**Table 13.3 Smart Card Interface Register Settings**

| | Bit | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Register** | **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** |
| SMR | GM | 0 | 1 | O/$\overline{E}$ | 1 | 0 | CKS1 | CKS0 |
| BRR | BRR7 | BRR6 | BRR5 | BRR4 | BRR3 | BRR2 | BRR1 | BRR0 |
| SCR | TIE | RIE | TE | RE | 0 | 0 | CKE1* | CKE0 |
| TDR | TDR7 | TDR6 | TDR5 | TDR4 | TDR3 | TDR2 | TDR1 | TDR0 |
| SSR | TDRE | RDRF | ORER | ERS | PER | TEND | 0 | 0 |
| RDR | RDR7 | RDR6 | RDR5 | RDR4 | RDR3 | RDR2 | RDR1 | RDR0 |
| SCMR | — | — | — | — | SDIR | SINV | — | SMIF |

Notes: —: Unused bit.
*: The CKE1 bit must be cleared to 0 when the GM bit in SMR is cleared to 0.

**SMR Setting:** The GM bit is cleared to 0 in normal smart card interface mode, and set to 1 in GSM mode. The O/$\overline{E}$ bit is cleared to 0 if the IC card is of the direct convention type, and set to 1 if of the inverse convention type.

Bits CKS1 and CKS0 select the clock source of the on-chip baud rate generator. See section 13.3.5, Clock.

**BRR Setting:** BRR is used to set the bit rate. See section 13.3.5, Clock, for the method of calculating the value to be set.

**SCR Setting:** The function of the TIE, RIE, TE, and RE bits is the same as for the normal SCI. For details, see section 12, Serial Communication Interface.

Bits CKE1 and CKE0 specify the clock output. When the GM bit in SMR is cleared to 0, set these bits to B'00 if a clock is not to be output, or to B'01 if a clock is to be output. When the GM bit in SMR is set to 1, clock output is performed. The clock output can also be fixed high or low.

**Smart Card Mode Register (SCMR) Setting:**

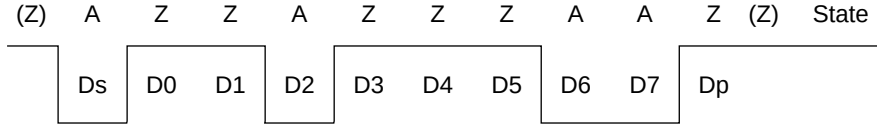
The SDIR bit is cleared to 0 if the IC card is of the direct convention type, and set to 1 if of the inverse convention type.

The SINV bit is cleared to 0 if the IC card is of the direct convention type, and set to 1 if of the inverse convention type.

The SMIF bit is set to 1 in the case of the Smart Card interface.

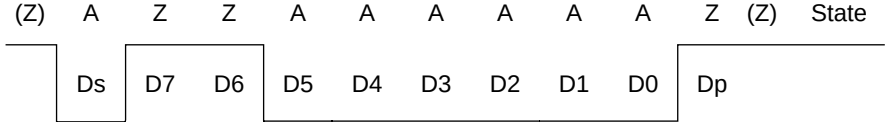
Examples of register settings and the waveform of the start character are shown below for the two types of IC card (direct convention and inverse convention).

- Direct convention (SDIR = SINV = O/$\overline{E}$ = 0)




With the direct convention type, the logic 1 level corresponds to state Z and the logic 0 level to state A, and transfer is performed in LSB-first order. The start character data above is H'3B. The parity bit is 1 since even parity is stipulated for the Smart Card.

- Inverse convention (SDIR = SINV = O/$\overline{E}$ = 1)




With the inverse convention type, the logic 1 level corresponds to state A and the logic 0 level to state Z, and transfer is performed in MSB-first order. The start character data above is H'3F. The parity bit is 0, corresponding to state Z, since even parity is stipulated for the Smart Card. With the H8S/2345 Series, inversion specified by the SINV bit applies only to the data bits, D7 to D0. For parity bit inversion, the O/$\overline{E}$ bit in SMR is set to odd parity mode (the same applies to both transmission and reception).

### 13.3.5 Clock

Only an internal clock generated by the on-chip baud rate generator can be used as the transmit/receive clock for the smart card interface. The bit rate is set with BRR and the CKS1 and CKS0 bits in SMR. The formula for calculating the bit rate is as shown below. Table 13.5 shows some sample bit rates.

If clock output is selected by setting CKE0 to 1, a clock with a frequency of 372 times the bit rate is output from the SCK pin.

$$B = \frac{\phi}{1488 \times 2^{2n-1} \times (N + 1)} \times 10^6$$

Where: N = Value set in BRR ($0 \leq N \leq 255$)
B = Bit rate (bit/s)
ø = Operating frequency (MHz)
n = See table 13.4

**Table 13.4 Correspondence between n and CKS1, CKS0**

| n | CKS1 | CKS0 |
|---|---|---|
| 0 | 0 | 0 |
| 1 | | 1 |
| 2 | 1 | 0 |
| 3 | | 1 |

**Table 13.5 Examples of Bit Rate B (bit/s) for Various BRR Settings (When n = 0)**

| | ø (MHz) | | | | | | |
|---|---|---|---|---|---|---|---|
| **N** | **10.00** | **10.714** | **13.00** | **14.285** | **16.00** | **18.00** | **20.00** |
| 0 | 13441 | 14400 | 17473 | 19200 | 21505 | 24194 | 26882 |
| 1 | 6720 | 7200 | 8737 | 9600 | 10753 | 12097 | 13441 |
| 2 | 4480 | 4800 | 5824 | 6400 | 7168 | 8065 | 8961 |

Note: Bit rates are rounded to the nearest whole number.

The method of calculating the value to be set in the bit rate register (BRR) from the operating frequency and bit rate, on the other hand, is shown below. N is an integer, $0 \leq N \leq 255$, and the smaller error is specified.

$$N = \frac{\phi}{1488 \times 2^{2n-1} \times B} \times 10^6 - 1$$

**Table 13.6 Examples of BRR Settings for Bit Rate B (bit/s) (When n = 0)**

| | ø (MHz) | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 7.1424 | | 10.00 | | 10.7136 | | 13.00 | | 14.2848 | | 16.00 | | 18.00 | | 20.00 | |
| bit/s | N | Error | N | Error | N | Error | N | Error | N | Error | N | Error | N | Error | N | Error |
| 9600 | 0 | 0.00 | 1 | 30 | 1 | 25 | 1 | 8.99 | 1 | 0.00 | 1 | 12.01 | 2 | 15.99 | 2 | 6.60 |

**Table 13.7 Maximum Bit Rate at Various Frequencies (Smart Card Interface Mode)**

| ø (MHz) | Maximum Bit Rate (bit/s) | N | n |
|---|---|---|---|
| 7.1424 | 9600 | 0 | 0 |
| 10.00 | 13441 | 0 | 0 |
| 10.7136 | 14400 | 0 | 0 |
| 13.00 | 17473 | 0 | 0 |
| 14.2848 | 19200 | 0 | 0 |
| 16.00 | 21505 | 0 | 0 |
| 18.00 | 24194 | 0 | 0 |
| 20.00 | 26882 | 0 | 0 |

The bit rate error is given by the following formula:

$$\text{Error (\%)} = \left( \frac{\phi}{1488 \times 2^{2n-1} \times B \times (N + 1)} \times 10^6 - 1 \right) \times 100$$

### 13.3.6 Data Transfer Operations

**Initialization:** Before transmitting and receiving data, initialize the SCI as described below. Initialization is also necessary when switching from transmit mode to receive mode, or vice versa.

[1] Clear the TE and RE bits in SCR to 0.

[2] Clear the error flags ERS, PER, and ORER in SSR to 0.

[3] Set the O/$\overline{E}$ bit and CKS1 and CKS0 bits in SMR. Clear the C/$\overline{A}$, CHR, and MP bits to 0, and set the STOP and PE bits to 1.

[4] Set the SMIF, SDIR, and SINV bits in SCMR.
When the SMIF bit is set to 1, the TxD and RxD pins are both switched from ports to SCI pins, and are placed in the high-impedance state.

[5] Set the value corresponding to the bit rate in BRR.

[6] Set the CKE0 bit in SCR. Clear the TIE, RIE, TE, RE, MPIE, TEIE and CKE1 bits to 0.
If the CKE0 bit is set to 1, the clock is output from the SCK pin.

[7] Wait at least one bit interval, then set the TIE, RIE, TE, and RE bits in SCR. Do not set the TE bit and RE bit at the same time, except for self-diagnosis.

**Serial Data Transmission:** As data transmission in smart card mode involves error signal sampling and retransmission processing, the processing procedure is different from that for the normal SCI. Figure 13.4 shows an example of the transmission processing flow.

Also, figure 13.5 shows the relationship between transmission operations and the internal registers.

[1] Perform Smart Card interface mode initialization as described above in Initialization.

[2] Check that the ERS error flag in SSR is cleared to 0.

[3] Repeat steps [2] and [3] until it can be confirmed that the TEND flag in SSR is set to 1.

[4] Write the transmit data to TDR, clear the TDRE flag to 0, and perform the transmit operation. The TEND flag is cleared to 0.

[5] When transmitting data continuously, go back to step [2].

[6] To end transmission, clear the TE bit to 0.
With the above processing, interrupt servicing or data transfer by the DTC is possible.

If transmission ends and the TEND flag is set to 1 while the TIE bit is set to 1 and interrupt requests are enabled, a transmit data empty interrupt (TXI) request will be generated. If an error occurs in transmission and the ERS flag is set to 1 while the RIE bit is set to 1 and interrupt requests are enabled, a transfer error interrupt (ERI) request will be generated.

The timing for setting the TEND flag depends on the value of the GM bit in SMR. The TEND timing is shown in figure 13.6.

If the DTC is activated by a TXI request, the number of bytes set in the DTC can be transmitted automatically, including automatic retransmission.

For details, see Interrupt Operations and Data Transfer Operation by DTC below.

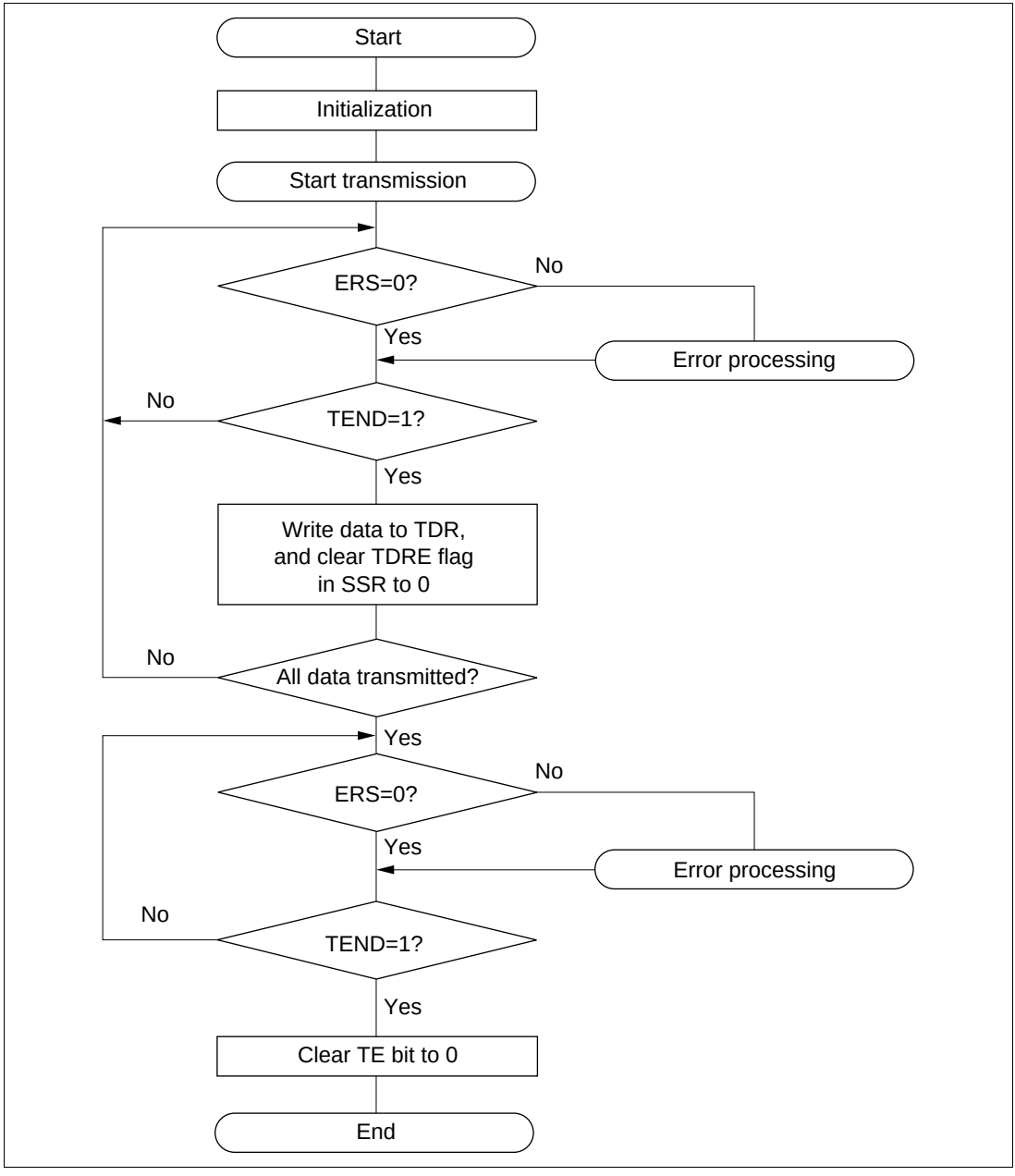



**Figure 13.4 Example of Transmission Processing Flow**

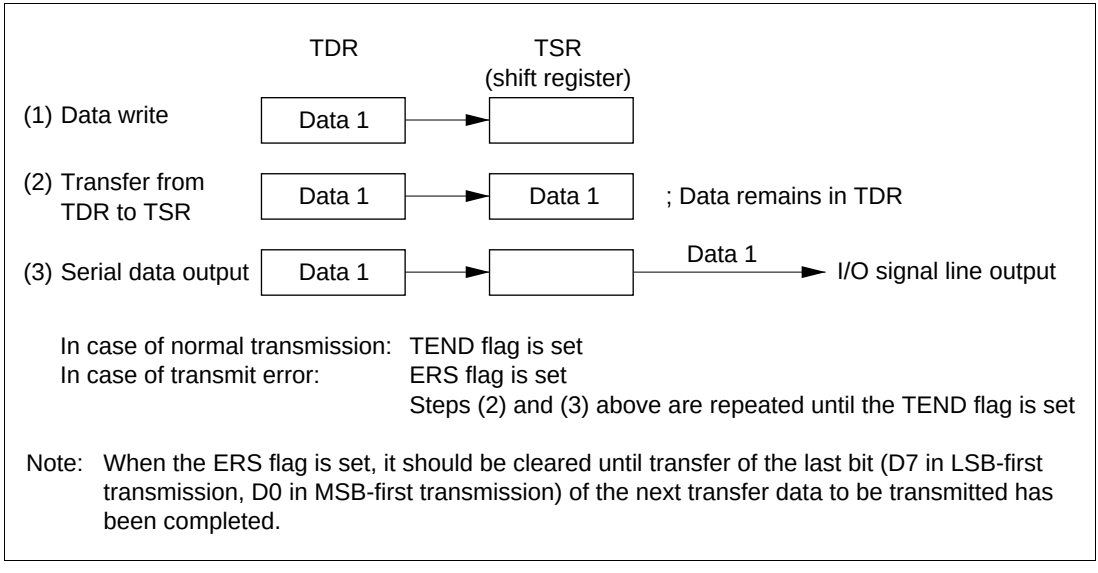



**Figure 13.5 Relation Between Transmit Operation and Internal Registers**

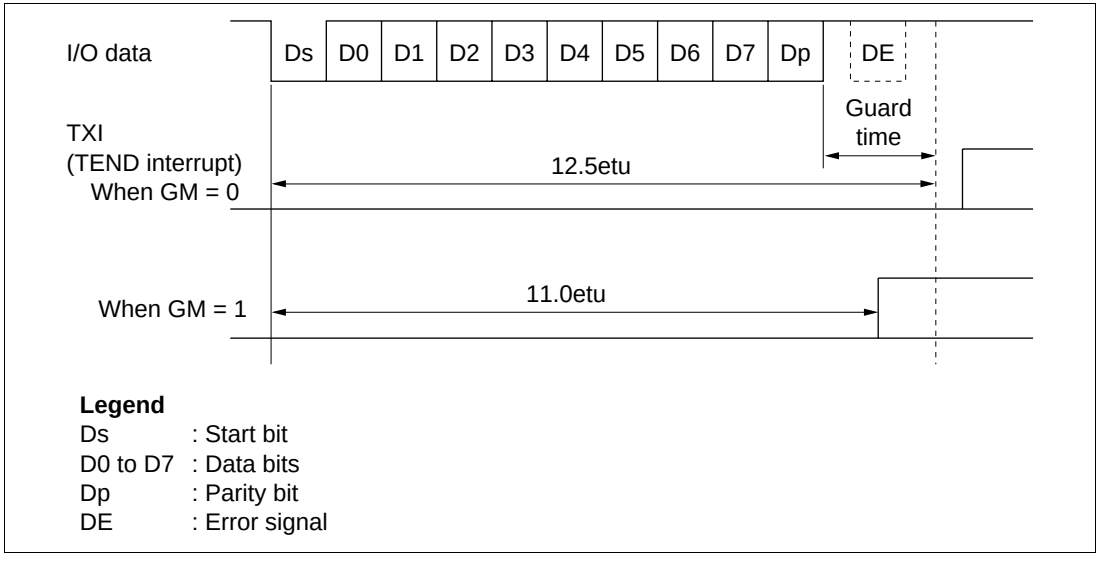



**Figure 13.6 TEND Flag Generation Timing in Transmission Operation**

**Serial Data Reception:** Data reception in Smart Card mode uses the same processing procedure as for the normal SCI. Figure 13.7 shows an example of the transmission processing flow.

[1] Perform Smart Card interface mode initialization as described above in Initialization.

[2] Check that the ORER flag and PER flag in SSR are cleared to 0. If either is set, perform the appropriate receive error processing, then clear both the ORER and the PER flag to 0.

[3] Repeat steps [2] and [3] until it can be confirmed that the RDRF flag is set to 1.

[4] Read the receive data from RDR.

[5] When receiving data continuously, clear the RDRF flag to 0 and go back to step [2].

[6] To end reception, clear the RE bit to 0.

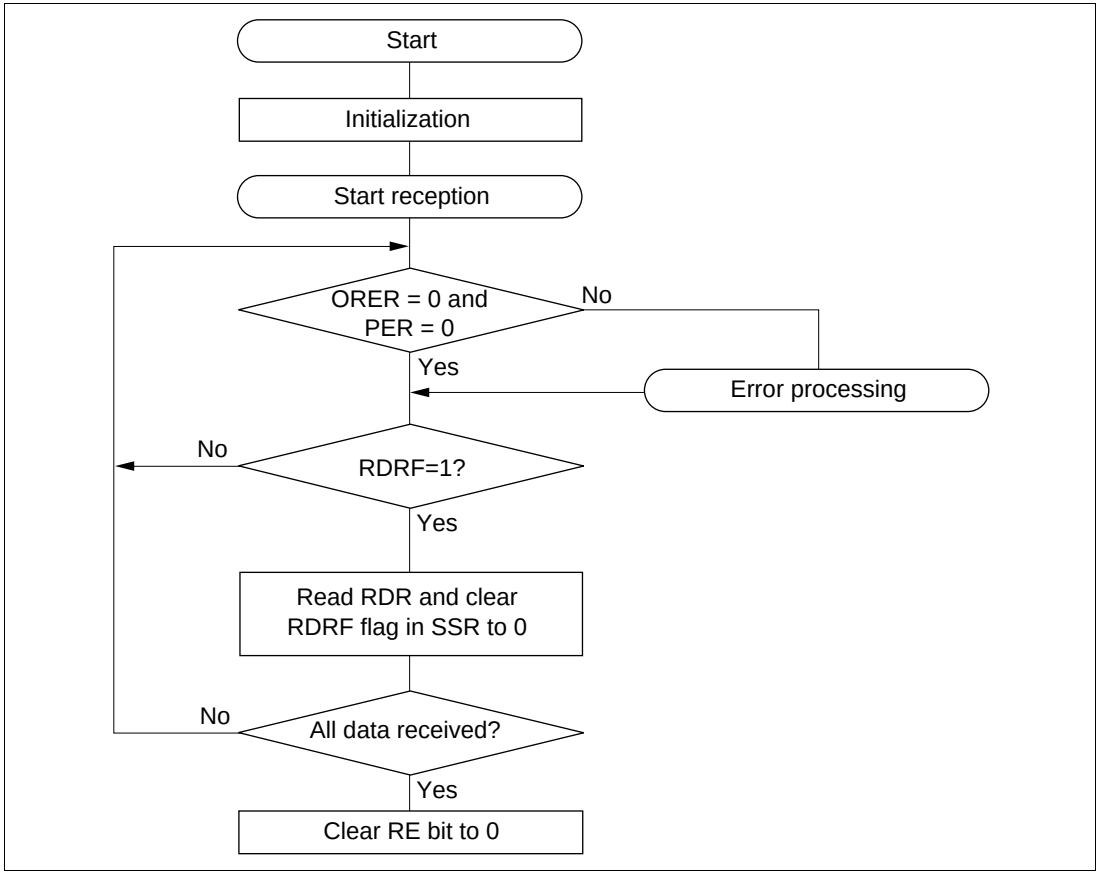



**Figure 13.7 Example of Reception Processing Flow**

With the above processing, interrupt servicing or data transfer by the DTC is possible.

If reception ends and the RDRF flag is set to 1 while the RIE bit is set to 1 and interrupt requests are enabled, a receive data full interrupt (RXI) request will be generated. If an error occurs in reception and either the ORER flag or the PER flag is set to 1, a transfer error interrupt (ERI) request will be generated.

If the DTC is activated by an RXI request, the receive data in which the error occurred is skipped, and only the number of bytes of receive data set in the DTC are transferred.

For details, see Interrupt Operation and Data Transfer Operation by DTC below.

If a parity error occurs during reception and the PER is set to 1, the received data is still transferred to RDR, and therefore this data can be read.

**Mode Switching Operation:** When switching from receive mode to transmit mode, first confirm that the receive operation has been completed, then start from initialization, clearing RE bit to 0 and setting TE bit to 1. The RDRF flag or the PER and ORER flags can be used to check that the receive operation has been completed.

When switching from transmit mode to receive mode, first confirm that the transmit operation has been completed, then start from initialization, clearing TE bit to 0 and setting RE bit to 1. The TEND flag can be used to check that the transmit operation has been completed.

**Fixing Clock Output Level:** When the GM bit in SMR is set to 1, the clock output level can be fixed with bits CKE1 and CKE0 in SCR. At this time, the minimum clock pulse width can be made the specified width.

Figure 13.8 shows the timing for fixing the clock output level. In this example, GM is set to 1, CKE1 is cleared to 0, and the CKE0 bit is controlled.

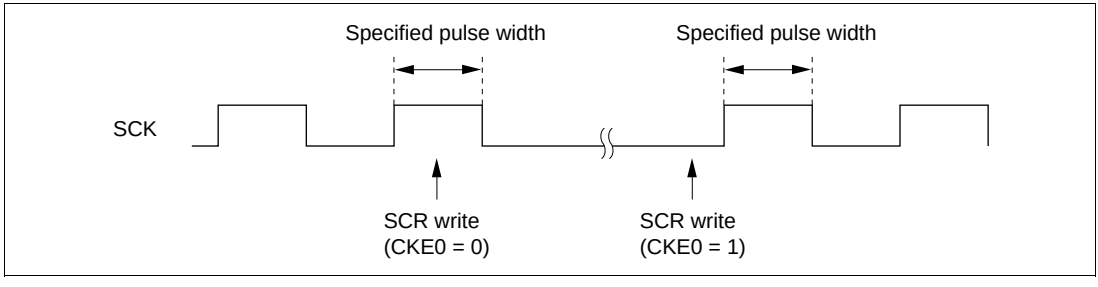



**Figure 13.8 Timing for Fixing Clock Output Level**

**Interrupt Operation:** There are three interrupt sources in smart card interface mode: transmit data empty interrupt (TXI) requests, transfer error interrupt (ERI) requests, and receive data full interrupt (RXI) requests. The transmit end interrupt (TEI) request is not used in this mode.

When the TEND flag in SSR is set to 1, a TXI interrupt request is generated.

When the RDRF flag in SSR is set to 1, an RXI interrupt request is generated.

When any of flags ORER, PER, and ERS in SSR is set to 1, an ERI interrupt request is generated. The relationship between the operating states and interrupt sources is shown in table 13.8.

**Table 13.8 Smart Card Mode Operating States and Interrupt Sources**

| Operating State | | Flag | Enable Bit | Interrupt Source | DTC Activation |
|---|---|---|---|---|---|
| Transmit Mode | Normal operation | TEND | TIE | TXI | Possible |
| | Error | ERS | RIE | ERI | Not possible |
| Receive Mode | Normal operation | RDRF | RIE | RXI | Possible |
| | Error | PER, ORER | RIE | ERI | Not possible |

**Data Transfer Operation by DTC:** In smart card mode, as with the normal SCI, transfer can be carried out using the DTC. In a transmit operation, the TDRE flag is also set to 1 at the same time as the TEND flag in SSR, and a TXI interrupt is generated. If the TXI request is designated beforehand as a DTC activation source, the DTC will be activated by the TXI request, and transfer of the transmit data will be carried out. The TDRE and TEND flags are automatically cleared to 0 when data transfer is performed by the DTC. In the event of an error, the SCI retransmits the same data automatically. However, the ERS flag is not cleared automatically when an error occurs, and so the RIE bit should be set to 1 beforehand so that an ERI request will be generated in the event of an error, and the ERS flag will be cleared.

When performing transfer using the DTC, it is essential to set and enable the DTC before carrying out SCI setting. For details of the DTC setting procedures, see section 8, Data Transfer Controller (DTC).

In a receive operation, an RXI interrupt request is generated when the RDRF flag in SSR is set to 1. If the RXI request is designated beforehand as a DTC activation source, the DTC will be activated by the RXI request, and transfer of the receive data will be carried out. The RDRF flag is cleared to 0 automatically when data transfer is performed by the DTC. If an error occurs, an error flag is set but the RDRF flag is not. The DTC is not activated, but instead, an ERI interrupt request is sent to the CPU. Therefore, the error flag should be cleared.

### 13.3.7 Operation in GSM Mode

**Switching the Mode:** When switching between smart card interface mode and software standby mode, the following switching procedure should be followed in order to maintain the clock duty.

- When changing from smart card interface mode to software standby mode

[1] Set the data register (DR) and data direction register (DDR) corresponding to the SCK pin to the value for the fixed output state in software standby mode.

[2] Write 0 to the TE bit and RE bit in the serial control register (SCR) to halt transmit/receive operation. At the same time, set the CKE1 bit to the value for the fixed output state in software standby mode.

[3] Write 0 to the CKE0 bit in SCR to halt the clock.

[4] Wait for one serial clock period.
During this interval, clock output is fixed at the specified level, with the duty preserved.

[5] Write H'00 to SMR and SCMR.

[6] Make the transition to the software standby state.

- When returning to smart card interface mode from software standby mode

[7] Exit the software standby state.

[8] Set the CKE1 bit in SCR to the value for the fixed output state (current SCK pin state) when software standby mode is initiated.

[9] Set smart card interface mode and output the clock. Signal generation is started with the normal duty.

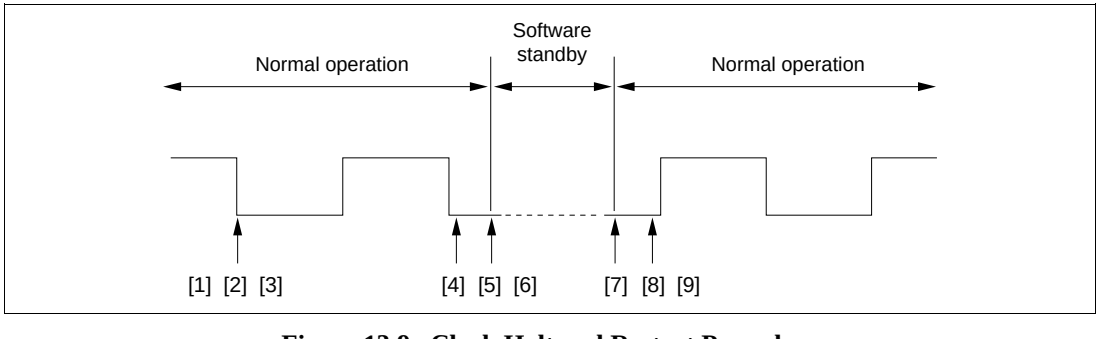



**Figure 13.9 Clock Halt and Restart Procedure**

**Powering On:** To secure the clock duty from power-on, the following switching procedure should be followed.

[1] The initial state is port input and high impedance. Use a pull-up resistor or pull-down resistor to fix the potential.

[2] Fix the SCK pin to the specified output level with the CKE1 bit in SCR.

[3] Set SMR and SCMR, and switch to smart card mode operation.

[4] Set the CKE0 bit in SCR to 1 to start clock output.

## 13.4 Usage Note

The following points should be noted when using the SCI as a smart card interface.

**Receive Data Sampling Timing and Reception Margin in Smart Card Interface Mode:** In smart card interface mode, the SCI operates on a basic clock with a frequency of 372 times the transfer rate.

In reception, the SCI samples the falling edge of the start bit using the basic clock, and performs internal synchronization. Receive data is latched internally at the rising edge of the 186th pulse of the basic clock. This is illustrated in figure 13.10.

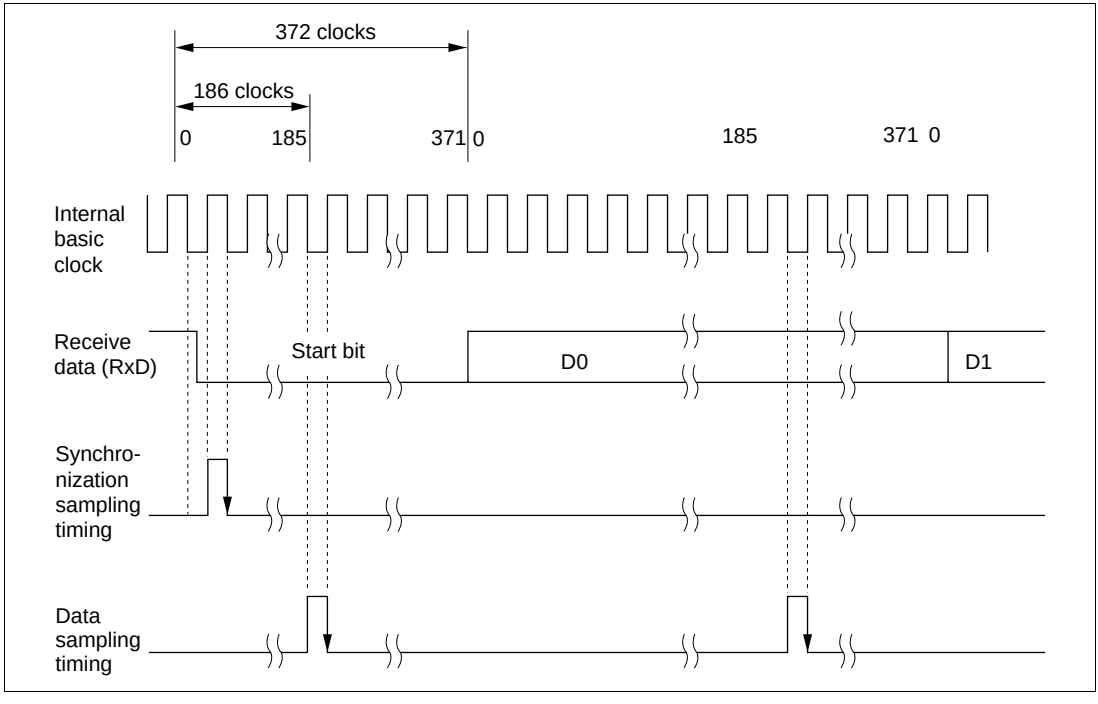



**Figure 13.10 Receive Data Sampling Timing in Smart Card Mode**

Thus the reception margin in smart card interface mode is given by the following formula.

$$M = \left| (0.5 - \frac{1}{2N}) - (L - 0.5)\,F - \frac{|D - 0.5|}{N}\,(1 + F) \right| \times 100\%$$

Where M: Reception margin (%)
N: Ratio of bit rate to clock (N = 372)
D: Clock duty (D = 0 to 1.0)
L: Frame length (L = 10)
F: Absolute value of clock frequency deviation

Assuming values of F = 0 and D = 0.5 in the above formula, the reception margin formula is as follows.

When D = 0.5 and F = 0,

$$M = (0.5 - 1/2 \times 372) \times 100\%$$
$$= 49.866\%$$

**Retransfer Operations:** Retransfer operations are performed by the SCI in receive mode and transmit mode as described below.

- Retransfer operation when SCI is in receive mode

  Figure 13.11 illustrates the retransfer operation when the SCI is in receive mode.

[1] If an error is found when the received parity bit is checked, the PER bit in SSR is automatically set to 1. If the RIE bit in SCR is enabled at this time, an ERI interrupt request is generated. The PER bit in SSR should be kept cleared to 0 until the next parity bit is sampled.

[2] The RDRF bit in SSR is not set for a frame in which an error has occurred.

[3] If no error is found when the received parity bit is checked, the PER bit in SSR is not set to 1.

[4] If no error is found when the received parity bit is checked, the receive operation is judged to have been completed normally, and the RDRF flag in SSR is automatically set to 1. If the RIE bit in SCR is enabled at this time, an RXI interrupt request is generated.

If DTC data transfer by an RXI source is enabled, the contents of RDR can be read automatically. When the RDR data is read by the DTC, the RDRF flag is automatically cleared to 0.

[5] When a normal frame is received, the pin retains the high-impedance state at the timing for error signal transmission.

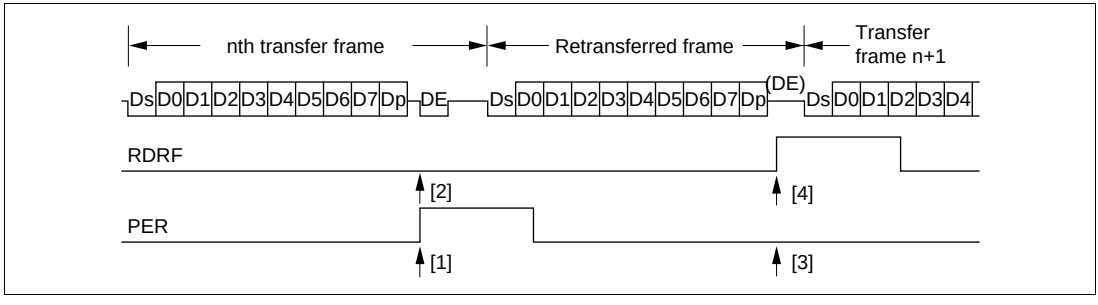



**Figure 13.11 Retransfer Operation in SCI Receive Mode**

- Retransfer operation when SCI is in transmit mode

  Figure 13.12 illustrates the retransfer operation when the SCI is in transmit mode.

[6] If an error signal is sent back from the receiving end after transmission of one frame is completed, the ERS bit in SSR is set to 1. If the RIE bit in SCR is enabled at this time, an ERI interrupt request is generated. The ERS bit in SSR should be kept cleared to 0 until the next parity bit is sampled.

[7] The TEND bit in SSR is not set for a frame for which an error signal indicating an abnormality is received.

[8] If an error signal is not sent back from the receiving end, the ERS bit in SSR is not set.

[9] If an error signal is not sent back from the receiving end, transmission of one frame, including a retransfer, is judged to have been completed, and the TEND bit in SSR is set to 1. If the TIE bit in SCR is enabled at this time, a TXI interrupt request is generated.

If data transfer by the DTC by means of the TXI source is enabled, the next data can be written to TDR automatically. When data is written to TDR by the DTC, the TDRE bit is automatically cleared to 0.

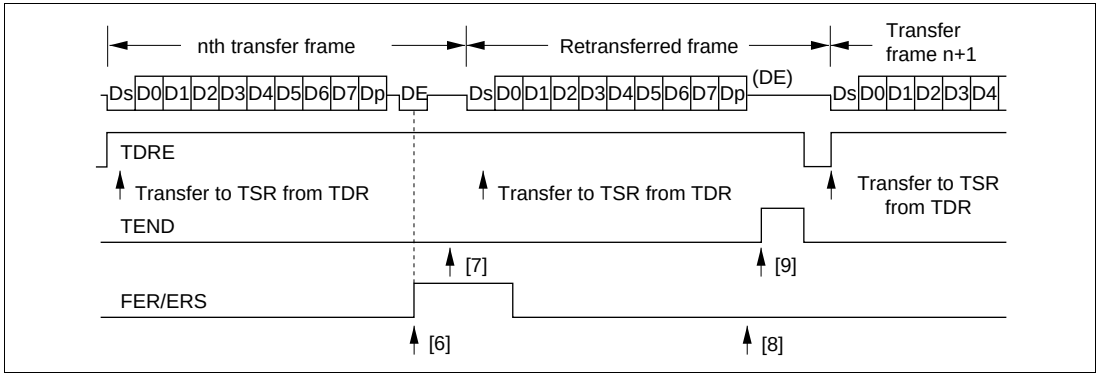



**Figure 13.12 Retransfer Operation in SCI Transmit Mode**

# Section 14 A/D Converter

## 14.1 Overview

The H8S/2345 Series incorporates a successive approximation type 10-bit A/D converter that allows up to eight analog input channels to be selected.

### 14.1.1 Features

A/D converter features are listed below

- 10-bit resolution
- Eight input channels
- Settable analog conversion voltage range
  - — Conversion of analog voltages with the reference voltage pin ($V_{ref}$) as the analog reference voltage
- High-speed conversion
  - — Minimum conversion time: 6.7 µs per channel (at 20 MHz operation)
- Choice of single mode or scan mode
  - — Single mode: Single-channel A/D conversion
  - — Scan mode: Continuous A/D conversion on 1 to 4 channels
- Four data registers
  - — Conversion results are held in a 16-bit data register for each channel
- Sample and hold function
- Three kinds of conversion start
  - — Choice of software or timer conversion start trigger (TPU or 8-bit timer), or $\overline{\text{ADTRG}}$ pin
- A/D conversion end interrupt generation
  - — A/D conversion end interrupt (ADI) request can be generated at the end of A/D conversion
- Module stop mode can be set
  - — As the initial setting, A/D converter operation is halted. Register access is enabled by exiting module stop mode.

### 14.1.2 Block Diagram

Figure 14.1 shows a block diagram of the A/D converter.

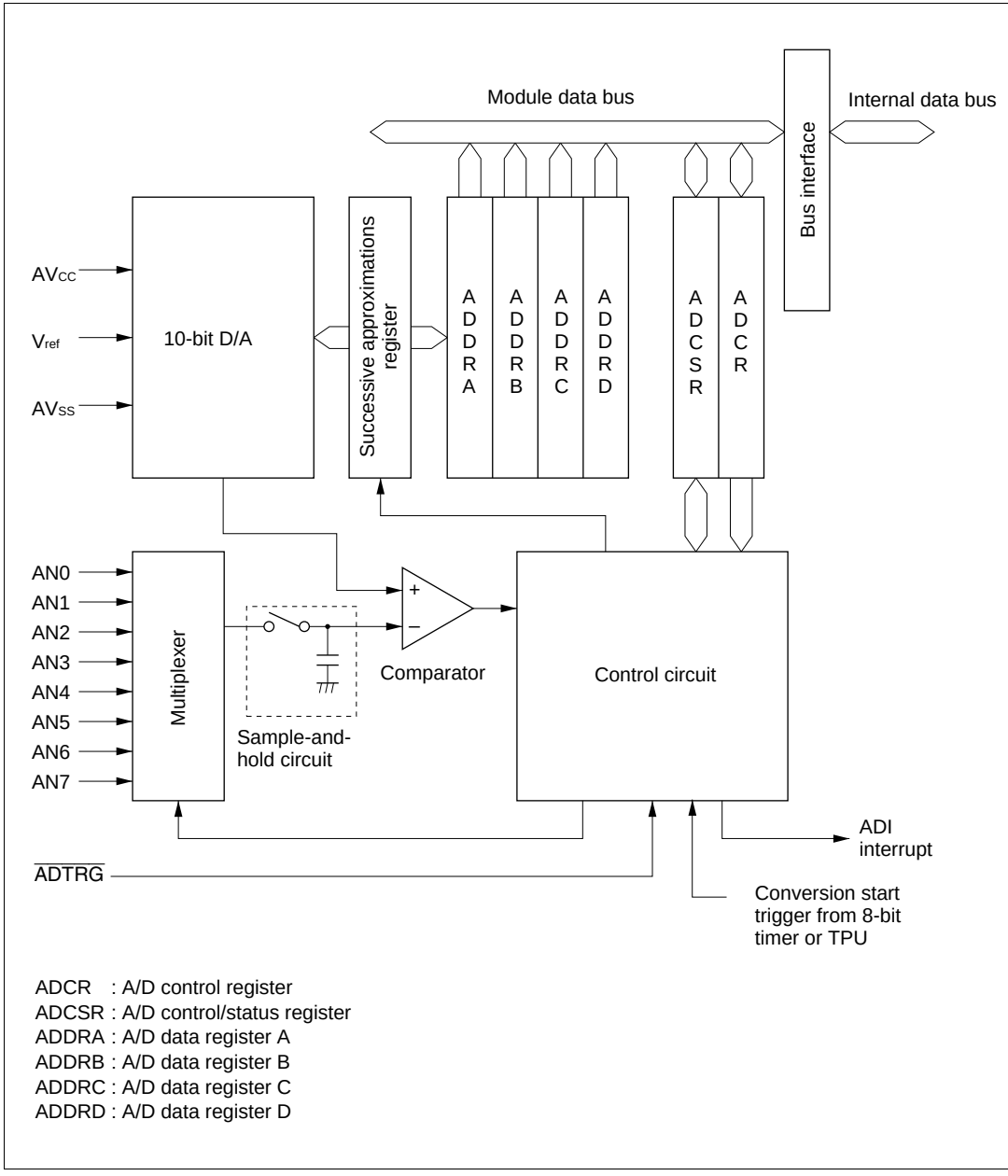



**Figure 14.1 Block Diagram of A/D Converter**

### 14.1.3 Pin Configuration

Table 14.1 summarizes the input pins used by the A/D converter.

The $AV_{CC}$ and $AV_{SS}$ pins are the power supply pins for the analog block in the A/D converter. The $V_{ref}$ pin is the A/D conversion reference voltage pin.

The eight analog input pins are divided into two groups: group 0 (AN0 to AN3), and group 1 (AN4 to AN7).

**Table 14.1 A/D Converter Pins**

| Pin Name | Symbol | I/O | Function |
|---|---|---|---|
| Analog power supply pin | $AV_{CC}$ | Input | Analog block power supply |
| Analog ground pin | $AV_{SS}$ | Input | Analog block ground and A/D conversion reference voltage |
| Reference voltage pin | $V_{ref}$ | Input | A/D conversion reference voltage |
| Analog input pin 0 | AN0 | Input | Group 0 analog inputs |
| Analog input pin 1 | AN1 | Input | |
| Analog input pin 2 | AN2 | Input | |
| Analog input pin 3 | AN3 | Input | |
| Analog input pin 4 | AN4 | Input | Group 1 analog inputs |
| Analog input pin 5 | AN5 | Input | |
| Analog input pin 6 | AN6 | Input | |
| Analog input pin 7 | AN7 | Input | |
| A/D external trigger input pin | $\overline{\text{ADTRG}}$ | Input | External trigger input for starting A/D conversion |

### 14.1.4 Register Configuration

Table 14.2 summarizes the registers of the A/D converter.

**Table 14.2 A/D Converter Registers**

| Name | Abbreviation | R/W | Initial Value | Address*[1] |
|---|---|---|---|---|
| A/D data register AH | ADDRAH | R | H'00 | H'FF90 |
| A/D data register AL | ADDRAL | R | H'00 | H'FF91 |
| A/D data register BH | ADDRBH | R | H'00 | H'FF92 |
| A/D data register BL | ADDRBL | R | H'00 | H'FF93 |
| A/D data register CH | ADDRCH | R | H'00 | H'FF94 |
| A/D data register CL | ADDRCL | R | H'00 | H'FF95 |
| A/D data register DH | ADDRDH | R | H'00 | H'FF96 |
| A/D data register DL | ADDRDL | R | H'00 | H'FF97 |
| A/D control/status register | ADCSR | R/(W)*[2] | H'00 | H'FF98 |
| A/D control register | ADCR | R/W | H'3F | H'FF99 |
| Module stop control register | MSTPCR | R/W | H'3FFF | H'FF3C |

Notes: 1. Lower 16 bits of the address.
2. Bit 7 can only be written with 0 for flag clearing.

## 14.2 Register Descriptions

### 14.2.1 A/D Data Registers A to D (ADDRA to ADDRD)

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | AD9 | AD8 | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 | — | — | — | — | — | — |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |

There are four 16-bit read-only ADDR registers, ADDRA to ADDRD, used to store the results of A/D conversion.

The 10-bit data resulting from A/D conversion is transferred to the ADDR register for the selected channel and stored there. The upper 8 bits of the converted data are transferred to the upper byte (bits 15 to 8) of ADDR, and the lower 2 bits are transferred to the lower byte (bits 7 and 6) and stored. Bits 5 to 0 are always read as 0.

The correspondence between the analog input channels and ADDR registers is shown in table 14.3.

ADDR can always be read by the CPU. The upper byte can be read directly, but for the lower byte, data transfer is performed via a temporary register (TEMP). For details, see section 14.3, Interface to Bus Master.

The ADDR registers are initialized to H'0000 by a reset, and in standby mode or module stop mode.

**Table 14.3 Analog Input Channels and Corresponding ADDR Registers**

| Analog Input Channel | | |
|---|---|---|
| **Group 0** | **Group 1** | **A/D Data Register** |
| AN0 | AN4 | ADDRA |
| AN1 | AN5 | ADDRB |
| AN2 | AN6 | ADDRC |
| AN3 | AN7 | ADDRD |

### 14.2.2 A/D Control/Status Register (ADCSR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | ADF | ADIE | ADST | SCAN | CKS | CH2 | CH1 | CH0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/(W)* | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note: * Only 0 can be written to bit 7, to clear this flag.

ADCSR is an 8-bit readable/writable register that controls A/D conversion operations and shows the status of the operation.

ADCSR is initialized to H'00 by a reset, and in hardware standby mode or module stop mode.

**Bit 7—A/D End Flag (ADF):** Status flag that indicates the end of A/D conversion.

| Bit 7 | | |
|---|---|---|
| **ADF** | **Description** | |
| 0 | [Clearing conditions] | (Initial value) |
| | • When 0 is written to the ADF flag after reading ADF = 1 | |
| | • When the DTC is activated by an ADI interrupt and ADDR is read | |
| 1 | [Setting conditions] | |
| | • Single mode: When A/D conversion ends | |
| | • Scan mode: When A/D conversion ends on all specified channels | |

**Bit 6—A/D Interrupt Enable (ADIE):** Selects enabling or disabling of interrupt (ADI) requests at the end of A/D conversion.

| Bit 6 | | |
|---|---|---|
| **ADIE** | **Description** | |
| 0 | A/D conversion end interrupt (ADI) request disabled | (Initial value) |
| 1 | A/D conversion end interrupt (ADI) request enabled | |

**Bit 5—A/D Start (ADST):** Selects starting or stopping on A/D conversion. Holds a value of 1 during A/D conversion.

The ADST bit can be set to 1 by software, a timer conversion start trigger, or the A/D external trigger input pin ($\overline{\text{ADTRG}}$).

**Bit 5**

| ADST | Description | |
|---|---|---|
| 0 | • A/D conversion stopped | (Initial value) |
| 1 | • Single mode: A/D conversion is started. Cleared to 0 automatically when conversion on the specified channel ends<br>• Scan mode: A/D conversion is started. Conversion continues sequentially on the selected channels until ADST is cleared to 0 by software, a reset, or a transition to standby mode or module stop mode. | |

**Bit 4—Scan Mode (SCAN):** Selects single mode or scan mode as the A/D conversion operating mode. See section 14.4, Operation, for single mode and scan mode operation. Only set the SCAN bit while conversion is stopped.

**Bit 4**

| SCAN | Description | |
|---|---|---|
| 0 | Single mode | (Initial value) |
| 1 | Scan mode | |

**Bit 3—Clock Select (CKS):** Sets the A/D conversion time. Only change the conversion time while ADST = 0.

**Bit 3**

| CKS | Description | |
|---|---|---|
| 0 | Conversion time = 266 states (max.) | (Initial value) |
| 1 | Conversion time = 134 states (max.) | |

**Bits 2 to 0—Channel Select 2 to 0 (CH2 to CH0):** Together with the SCAN bit, these bits select the analog input channels.

Only set the input channel while conversion is stopped.

| Group Selection | Channel Selection | | Description | |
|---|---|---|---|---|
| CH2 | CH1 | CH2 | Single Mode | Scan Mode |
| 0 | 0 | 0 | AN0 (Initial value) | AN0 |
| | | 1 | AN1 | AN0, AN1 |
| | 1 | 0 | AN2 | AN0 to AN2 |
| | | 1 | AN3 | AN0 to AN3 |
| 1 | 0 | 0 | AN4 | AN4 |
| | | 1 | AN5 | AN4, AN5 |
| | 1 | 0 | AN6 | AN4 to AN6 |
| | | 1 | AN7 | AN4 to AN7 |

### 14.2.3 A/D Control Register (ADCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | TRGS1 | TRGS0 | — | — | — | — | — | — |
| Initial value | : | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | — | — | R/W | — | — | — |

ADCR is an 8-bit readable/writable register that enables or disables external triggering of A/D conversion operations.

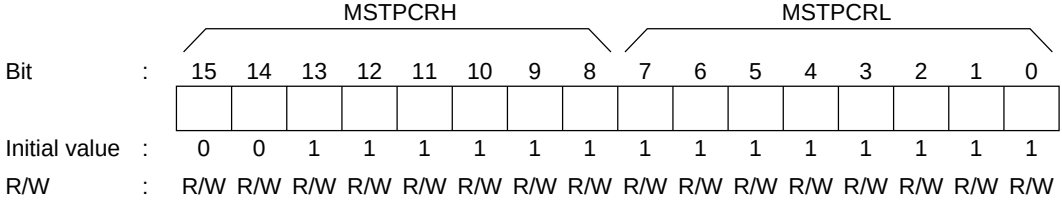
ADCR is initialized to H'3F by a reset, and in standby mode or module stop mode.

**Bits 7 and 6—Timer Trigger Select 1 and 0 (TRGS1, TRGS0):** Select enabling or disabling of the start of A/D conversion by a trigger signal. Only set bits TRGS1 and TRGS0 while conversion is stopped.

| Bit 7 | Bit 6 | | |
|---|---|---|---|
| TRGS1 | TRGS0 | Description | |
| 0 | 0 | A/D conversion start by external trigger is disabled | (Initial value) |
| | 1 | A/D conversion start by external trigger (TPU) is enabled | |
| 1 | 0 | A/D conversion start by external trigger (8-bit timer) is enabled | |
| | 1 | A/D conversion start by external trigger pin ($\overline{\text{ADTRG}}$) is enabled | |

**Bits 5 to 0—Reserved:** These bits are reserved; write as 1 in a write.

### 14.2.4 Module Stop Control Register (MSTPCR)




MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the MSTP9 bit in MSTPCR is set to 1, A/D converter operation stops at the end of the bus cycle and a transition is made to module stop mode. Registers cannot be read or written to in module stop mode. For details, see section 19.5, Module Stop Mode.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 9—Module Stop (MSTP9):** Specifies the A/D converter module stop mode.

**Bit 9**

| MSTP9 | Description | |
|---|---|---|
| 0 | A/D converter module stop mode cleared | |
| 1 | A/D converter module stop mode set | (Initial value) |

## 14.3 Interface to Bus Master

ADDRA to ADDRD are 16-bit registers, and the data bus to the bus master is 8 bits wide. Therefore, in accesses by the bus master, the upper byte is accessed directly, but the lower byte is accessed via a temporary register (TEMP).

A data read from ADDR is performed as follows. When the upper byte is read, the upper byte value is transferred to the CPU and the lower byte value is transferred to TEMP. Next, when the lower byte is read, the TEMP contents are transferred to the CPU.

When reading ADDR. always read the upper byte before the lower byte. It is possible to read only the upper byte, but if only the lower byte is read, incorrect data may be obtained.

Figure 14.2 shows the data flow for ADDR access.

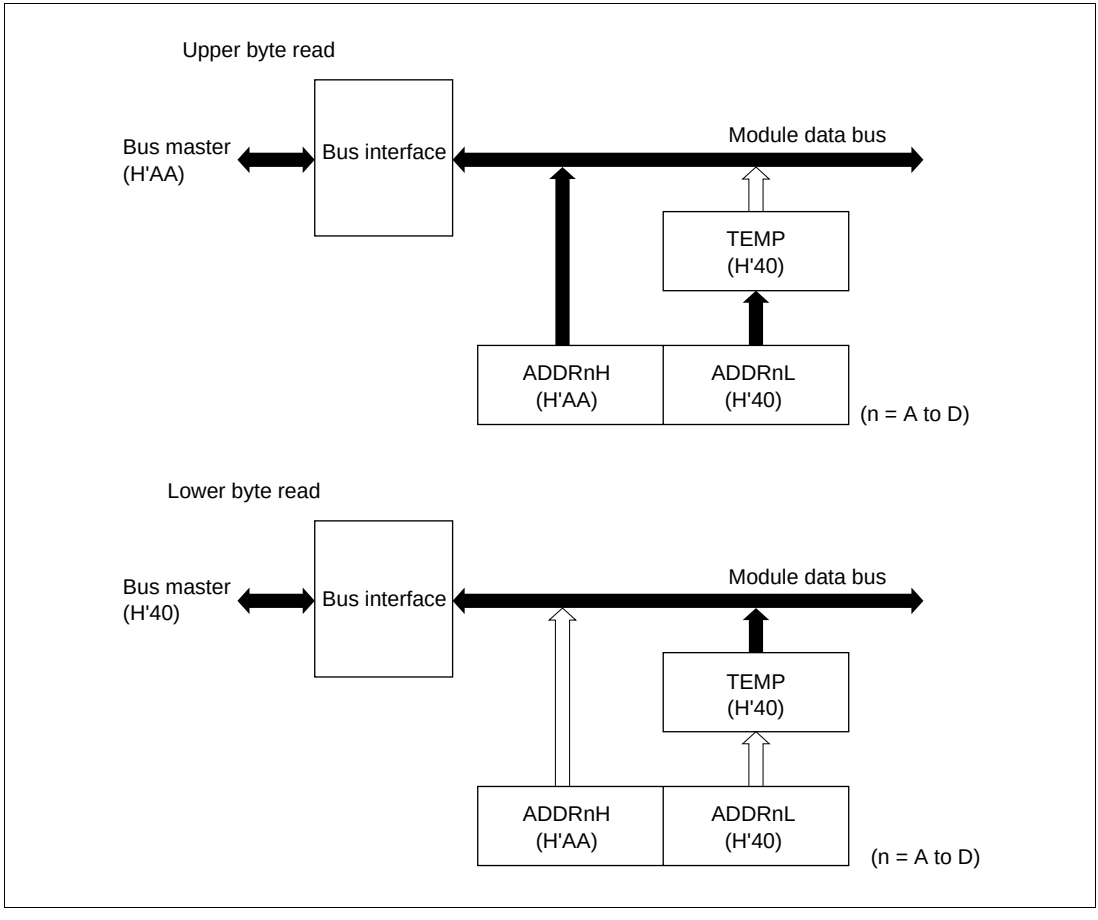



**Figure 14.2 ADDR Access Operation (Reading H'AA40)**

## 14.4 Operation

The A/D converter operates by successive approximation with 10-bit resolution. It has two operating modes: single mode and scan mode.

### 14.4.1 Single Mode (SCAN = 0)

Single mode is selected when A/D conversion is to be performed on a single channel only. A/D conversion is started when the ADST bit is set to 1, according to the software or external trigger input. The ADST bit remains set to 1 during A/D conversion, and is automatically cleared to 0 when conversion ends.

On completion of conversion, the ADF flag is set to 1. If the ADIE bit is set to 1 at this time, an ADI interrupt request is generated. The ADF flag is cleared by writing 0 after reading ADCSR.

When the operating mode or analog input channel must be changed during analog conversion, to prevent incorrect operation, first clear the ADST bit to 0 in ADCSR to halt A/D conversion. After making the necessary changes, set the ADST bit to 1 to start A/D conversion again. The ADST bit can be set at the same time as the operating mode or input channel is changed.

Typical operations when channel 1 (AN1) is selected in single mode are described next. Figure 14.3 shows a timing diagram for this example.

[1] Single mode is selected (SCAN = 0), input channel AN1 is selected (CH2 = 0, CH1 = 0, CH0 = 1), the A/D interrupt is enabled (ADIE = 1), and A/D conversion is started (ADST = 1).

[2] When A/D conversion is completed, the result is transferred to ADDRB. At the same time the ADF flag is set to 1, the ADST bit is cleared to 0, and the A/D converter becomes idle.

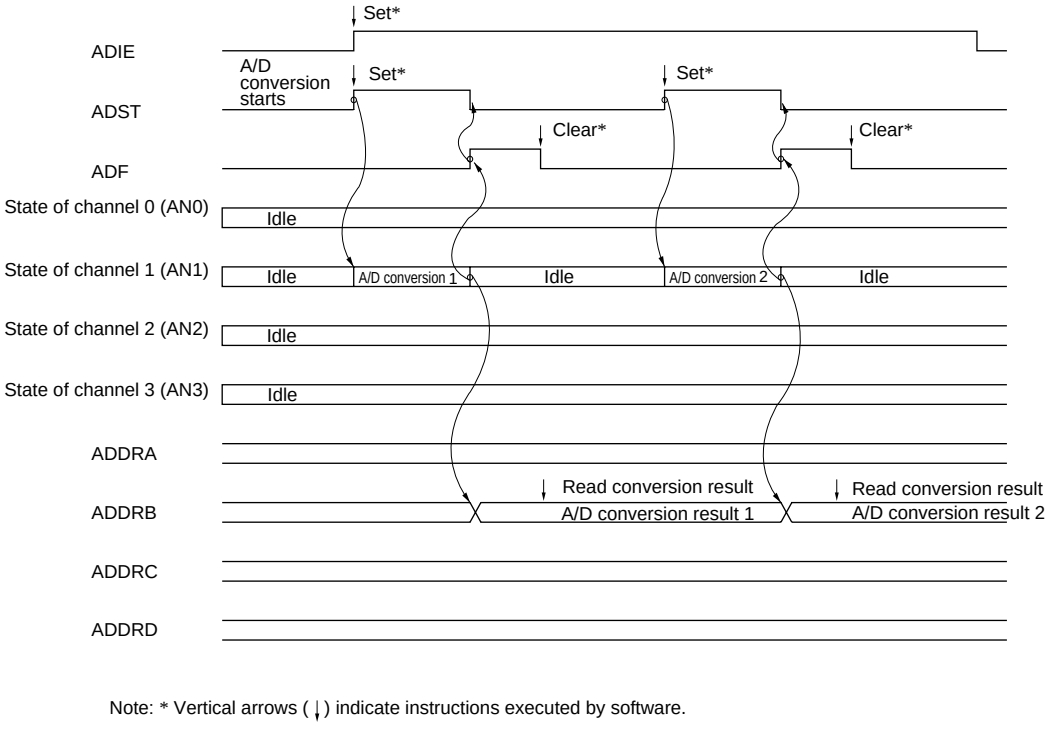
[3] Since ADF = 1 and ADIE = 1, an ADI interrupt is requested.

[4] The A/D interrupt handling routine starts.

[5] The routine reads ADCSR, then writes 0 to the ADF flag.

[6] The routine reads and processes the connection result (ADDRB).

[7] Execution of the A/D interrupt handling routine ends. After that, if the ADST bit is set to 1, A/D conversion starts again and steps [2] to [7] are repeated.




Note: * Vertical arrows ( ↓ ) indicate instructions executed by software.

**Figure 14.3 Example of A/D Converter Operation (Single Mode, Channel 1 Selected)**

### 14.4.2 Scan Mode (SCAN = 1)

Scan mode is useful for monitoring analog inputs in a group of one or more channels. When the ADST bit is set to 1 by a software, timer or external trigger input, A/D conversion starts on the first channel in the group (AN0). When two or more channels are selected, after conversion of the first channel ends, conversion of the second channel (AN1) starts immediately. A/D conversion continues cyclically on the selected channels until the ADST bit is cleared to 0. The conversion results are transferred for storage into the ADDR registers corresponding to the channels.

When the operating mode or analog input channel must be changed during analog conversion, to prevent incorrect operation, first clear the ADST bit to 0 in ADCSR to halt A/D conversion. After making the necessary changes, set the ADST bit to 1 to start A/D conversion again. The ADST bit can be set at the same time as the operating mode or input channel is changed.

Typical operations when three channels (AN0 to AN2) are selected in scan mode are described next. Figure 14.4 shows a timing diagram for this example.

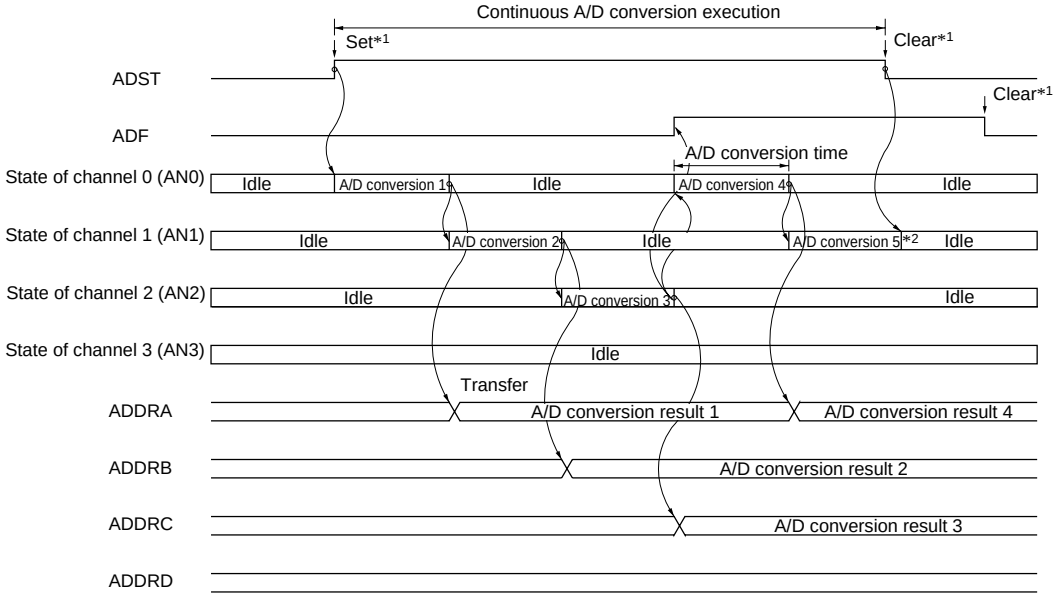
[1] Scan mode is selected (SCAN = 1), scan group 0 is selected (CH2 = 0), analog input channels AN0 to AN2 are selected (CH1 = 1, CH0 = 0), and A/D conversion is started (ADST = 1)

[2] When A/D conversion of the first channel (AN0) is completed, the result is transferred to ADDRA. Next, conversion of the second channel (AN1) starts automatically.

[3] Conversion proceeds in the same way through the third channel (AN2).

[4] When conversion of all the selected channels (AN0 to AN2) is completed, the ADF flag is set to 1 and conversion of the first channel (AN0) starts again. If the ADIE bit is set to 1 at this time, an ADI interrupt is requested after A/D conversion ends.

[5] Steps [2] to [4] are repeated as long as the ADST bit remains set to 1. When the ADST bit is cleared to 0, A/D conversion stops. After that, if the ADST bit is set to 1, A/D conversion starts again from the first channel (AN0).




Notes: 1. Vertical arrows (↓) indicate instructions executed by software.
2. Data currently being converted is ignored.

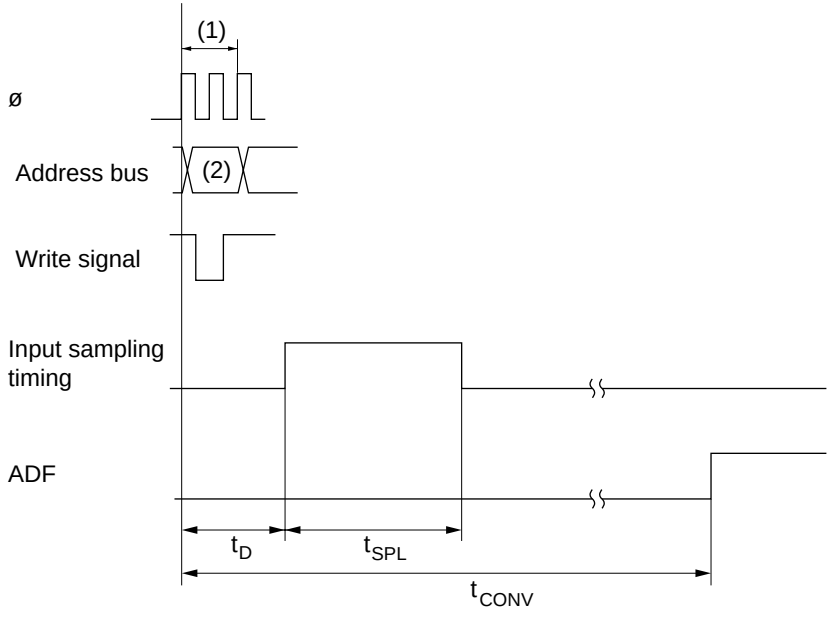
**Figure 14.4 Example of A/D Converter Operation (Scan Mode, Channels AN0 to AN2 Selected)**

### 14.4.3 Input Sampling and A/D Conversion Time

The A/D converter has a built-in sample-and-hold circuit. The A/D converter samples the analog input at a time $t_D$ after the ADST bit is set to 1, then starts conversion. Figure 14.5 shows the A/D conversion timing. Table 14.4 indicates the A/D conversion time.

As indicated in figure 14.5, the A/D conversion time includes $t_D$ and the input sampling time. The length of $t_D$ varies depending on the timing of the write access to ADCSR. The total conversion time therefore varies within the ranges indicated in table 14.4.

In scan mode, the values given in table 14.4 apply to the first conversion time. In the second and subsequent conversions the conversion time is fixed at 256 states when CKS = 0 or 128 states when CKS = 1.




Legend
(1) : ADCSR write cycle
(2) : ADCSR address
$t_D$ : A/D conversion start delay
$t_{SPL}$ : Input sampling time
$t_{CONV}$ : A/D conversion time

**Figure 14.5 A/D Conversion Timing**

**Table 14.4 A/D Conversion Time (Single Mode)**

| | | CKS = 0 | | | CKS = 1 | | |
|---|---|---|---|---|---|---|---|
| **Item** | **Symbol** | **Min** | **Typ** | **Max** | **Min** | **Typ** | **Max** |
| A/D conversion start delay | $t_D$ | 10 | — | 17 | 6 | — | 9 |
| Input sampling time | $t_{SPL}$ | — | 63 | — | — | 31 | — |
| A/D conversion time | $t_{CONV}$ | 259 | — | 266 | 131 | — | 134 |

Note: Values in the table are the number of states.

### 14.4.4 External Trigger Input Timing

A/D conversion can be externally triggered. When the TRGS1 and TRGS0 bits are set to 11 in ADCR, external trigger input is enabled at the $\overline{\text{ADTRG}}$ pin. A falling edge at the $\overline{\text{ADTRG}}$ pin sets the ADST bit to 1 in ADCSR, starting A/D conversion. Other operations, in both single and scan modes, are the same as if the ADST bit has been set to 1 by software. Figure 14.6 shows the timing.

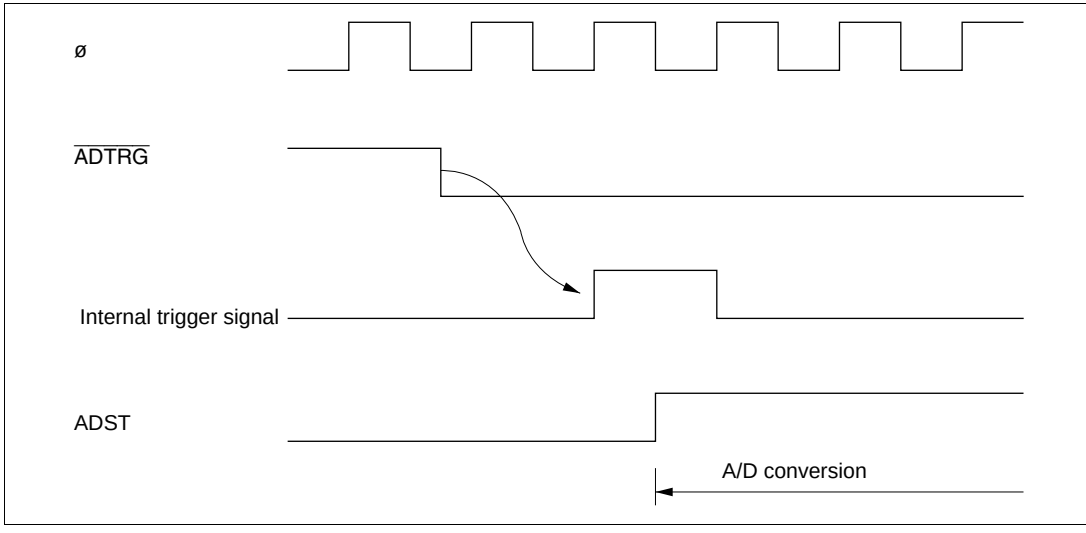



**Figure 14.6 External Trigger Input Timing**

## 14.5 Interrupts

The A/D converter generates an A/D conversion end interrupt (ADI) at the end of A/D conversion. ADI interrupt requests can be enabled or disabled by means of the ADIE bit in ADCSR.

The DTC can be activated by an ADI interrupt. Having the converted data read by the DTC in response to an ADI interrupt enables continuous conversion to be achieved without imposing a load on software.

The A/D converter interrupt source is shown in table 14.5.

**Table 14.5 A/D Converter Interrupt Source**

| Interrupt Source | Description | DTC Activation |
|---|---|---|
| ADI | Interrupt due to end of conversion | Possible |

## 14.6 Usage Notes

The following points should be noted when using the A/D converter.

**Setting Range of Analog Power Supply and Other Pins:**

(1) Analog input voltage range

The voltage applied to analog input pins AN0 to AN7 during A/D conversion should be in the range $AV_{SS} \leq ANn \leq V_{ref}$.

(2) Relation between $AV_{CC}$, $AV_{SS}$ and $V_{CC}$, $V_{SS}$

As the relationship between $AV_{CC}$, $AV_{SS}$ and $V_{CC}$, $V_{SS}$, set, $AV_{CC} = V_{CC}$ and $AV_{SS} = V_{SS}$. If the A/D converter is not used, the $AV_{CC}$ and $AV_{SS}$ pins must on no account be left open.

(3) $V_{ref}$ input range

The analog reference voltage input at the $V_{ref}$ pin set in the range $V_{ref} \leq AV_{CC}$.

If conditions (1), (2), and (3) above are not met, the reliability of the device may be adversely affected.

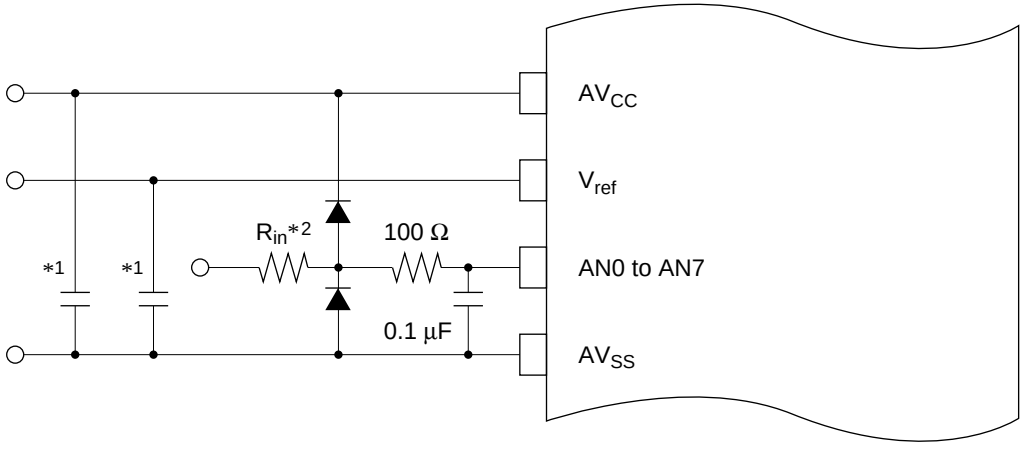
**Notes on Board Design:** In board design, digital circuitry and analog circuitry should be as mutually isolated as possible, and layout in which digital circuit signal lines and analog circuit signal lines cross or are in close proximity should be avoided as far as possible. Failure to do so may result in incorrect operation of the analog circuitry due to inductance, adversely affecting A/D conversion values.

Also, digital circuitry must be isolated from the analog input signals (AN0 to AN7), analog reference power supply ($V_{ref}$), and analog power supply ($AV_{CC}$) by the analog ground ($AV_{SS}$). Also, the analog ground ($AV_{SS}$) should be connected at one point to a stable digital ground ($V_{SS}$) on the board.

**Notes on Noise Countermeasures:** A protection circuit connected to prevent damage due to an abnormal voltage such as an excessive surge at the analog input pins (AN0 to AN7) and analog reference power supply ($V_{ref}$) should be connected between $AV_{CC}$ and $AV_{SS}$ as shown in figure 14.7.

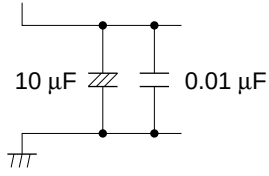
Also, the bypass capacitors connected to $AV_{CC}$ and $V_{ref}$ and the filter capacitor connected to AN0 to AN7 must be connected to $AV_{SS}$.

If a filter capacitor is connected as shown in figure 14.7, the input currents at the analog input pins (AN0 to AN7) are averaged, and so an error may arise. Also, when A/D conversion is performed frequently, as in scan mode, if the current charged and discharged by the capacitance of the sample-and-hold circuit in the A/D converter exceeds the current input via the input impedance ($R_{in}$), an error will arise in the analog input pin voltage. Careful consideration is therefore required when deciding the circuit constants.




Notes: Values are reference values.

1.



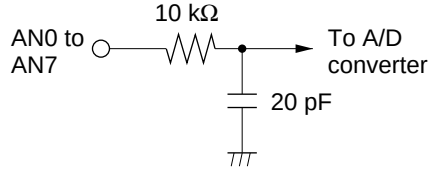

2. $R_{in}$: Input impedance

**Figure 14.7 Example of Analog Input Protection Circuit**

**Table 14.6 Analog Pin Specifications**

| Item | Min | Max | Unit |
|---|---|---|---|
| Analog input capacitance | — | 20 | pF |
| Permissible signal source impedance | — | 10* | kΩ |

Note: * When $V_{CC}$ = 4.0 V to 5.5 V and ø ≤ 12 MHz




Note: Values are reference values.

**Figure 14.8 Analog Input Pin Equivalent Circuit**

**A/D Conversion Precision Definitions:** H8S/2345 Series A/D conversion precision definitions are given below.

- Resolution

  The number of A/D converter digital output codes

- Offset error

  The deviation of the analog input voltage value from the ideal A/D conversion characteristic when the digital output changes from the minimum voltage value B'0000000000 (H'000) to B'0000000001 (H'001) (see figure 14.10).

- Full-scale error

  The deviation of the analog input voltage value from the ideal A/D conversion characteristic when the digital output changes from B'1111111110 (H'3FE) to B'1111111111 (H'3FF) (see figure 14.10).

- Quantization error

  The deviation inherent in the A/D converter, given by 1/2 LSB (see figure 14.9).

- Nonlinearity error

  The error with respect to the ideal A/D conversion characteristic between the zero voltage and the full-scale voltage. Does not include the offset error, full-scale error, or quantization error.

- Absolute precision

  The deviation between the digital value and the analog input value. Includes the offset error, full-scale error, quantization error, and nonlinearity error.

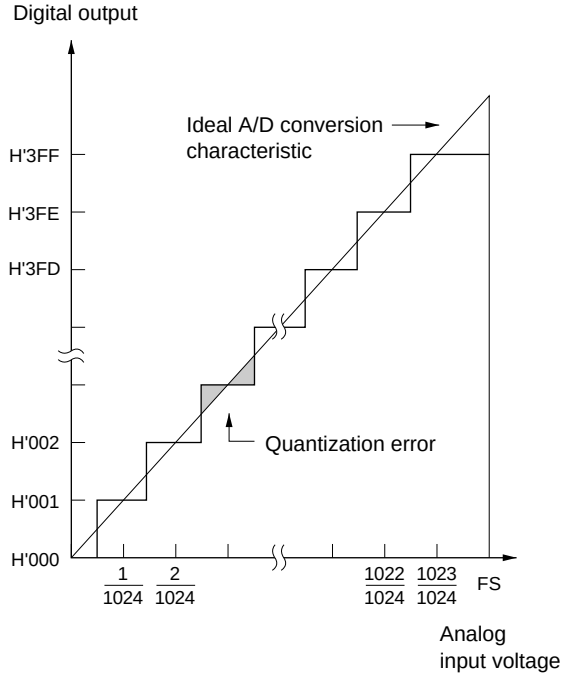



**Figure 14.9 A/D Conversion Precision Definitions (1)**

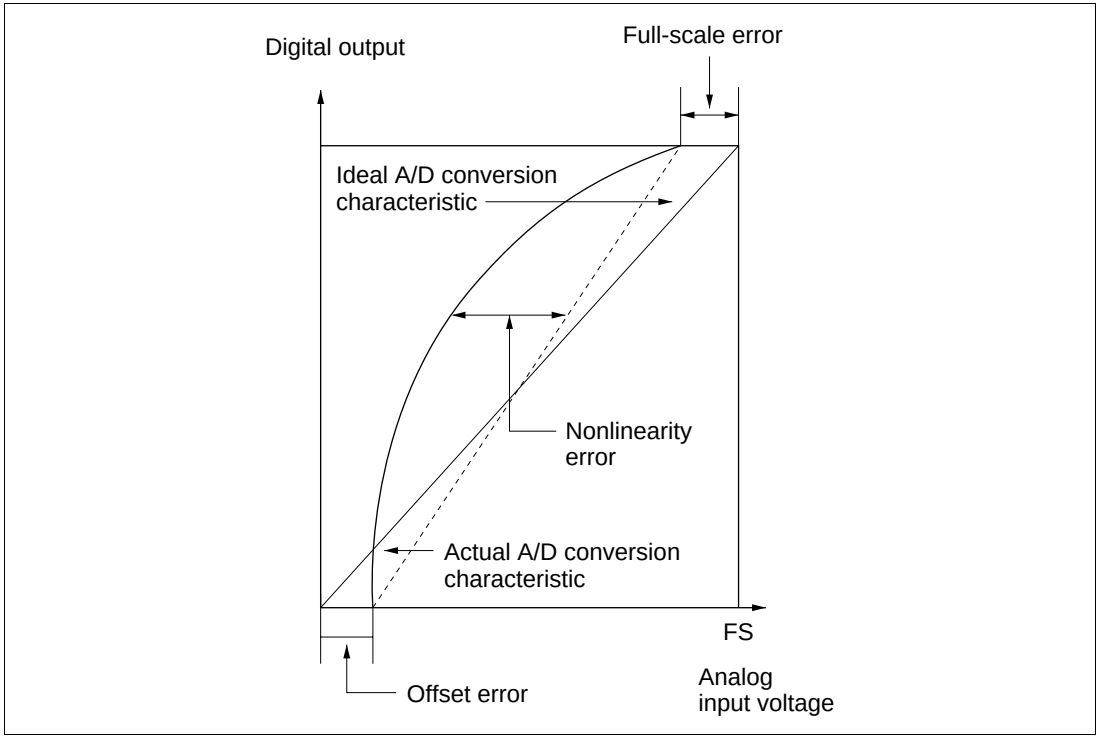



**Figure 14.10 A/D Conversion Precision Definitions (2)**

**Permissible Signal Source Impedance:** H8S/2345 Series analog input is designed so that conversion precision is guaranteed for an input signal for which the signal source impedance is 10 kΩ or less. This specification is provided to enable the A/D converter's sample-and-hold circuit input capacitance to be charged within the sampling time; if the sensor output impedance exceeds 10 kΩ, charging may be insufficient and it may not be possible to guarantee the A/D conversion precision.

However, if a large capacitance is provided externally, the input load will essentially comprise only the internal input resistance of 10 kΩ, and the signal source impedance is ignored.

However, since a low-pass filter effect is obtained in this case, it may not be possible to follow an analog signal with a large differential coefficient (e.g., 5 mV/μs or greater).

When converting a high-speed analog signal, a low-impedance buffer should be inserted.

**Influences on Absolute Precision:** Adding capacitance results in coupling with GND, and therefore noise in GND may adversely affect absolute precision. Be sure to make the connection to an electrically stable GND such as $AV_{SS}$.

Care is also required to insure that filter circuits do not communicate with digital signals on the mounting board, so acting as antennas.

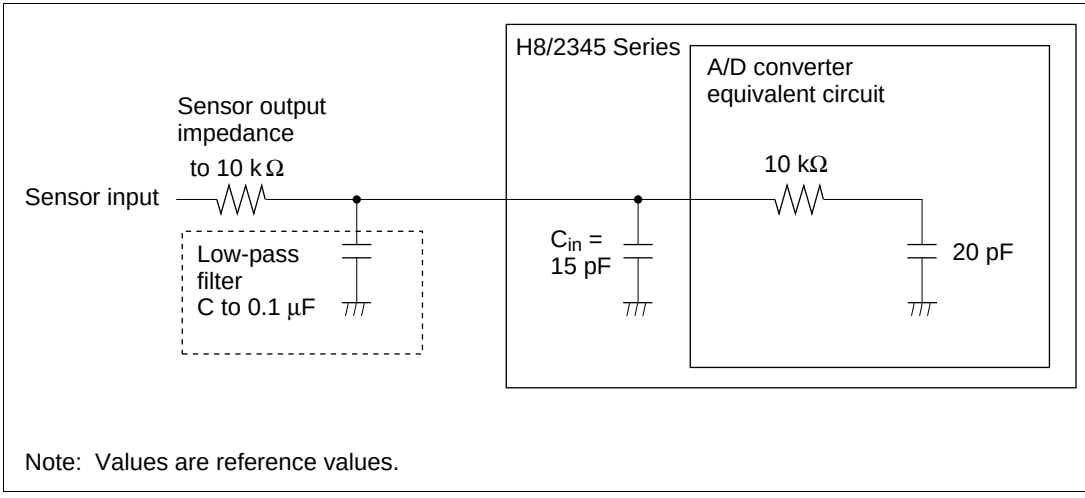



**Figure 14.11 Example of Analog Input Circuit**

# Section 15 D/A Converter

## 15.1 Overview

The H8S/2345 Series includes a two-channel D/A converter.

### 15.1.1 Features

D/A converter features are listed below

- 8-bit resolution
- Two output channels
- Maximum conversion time of 10 μs (with 20 pF load)
- Output voltage of 0 V to $V_{ref}$
- D/A output hold function in software standby mode

### 15.1.2 Block Diagram

Figure 15.1 shows a block diagram of the D/A converter.

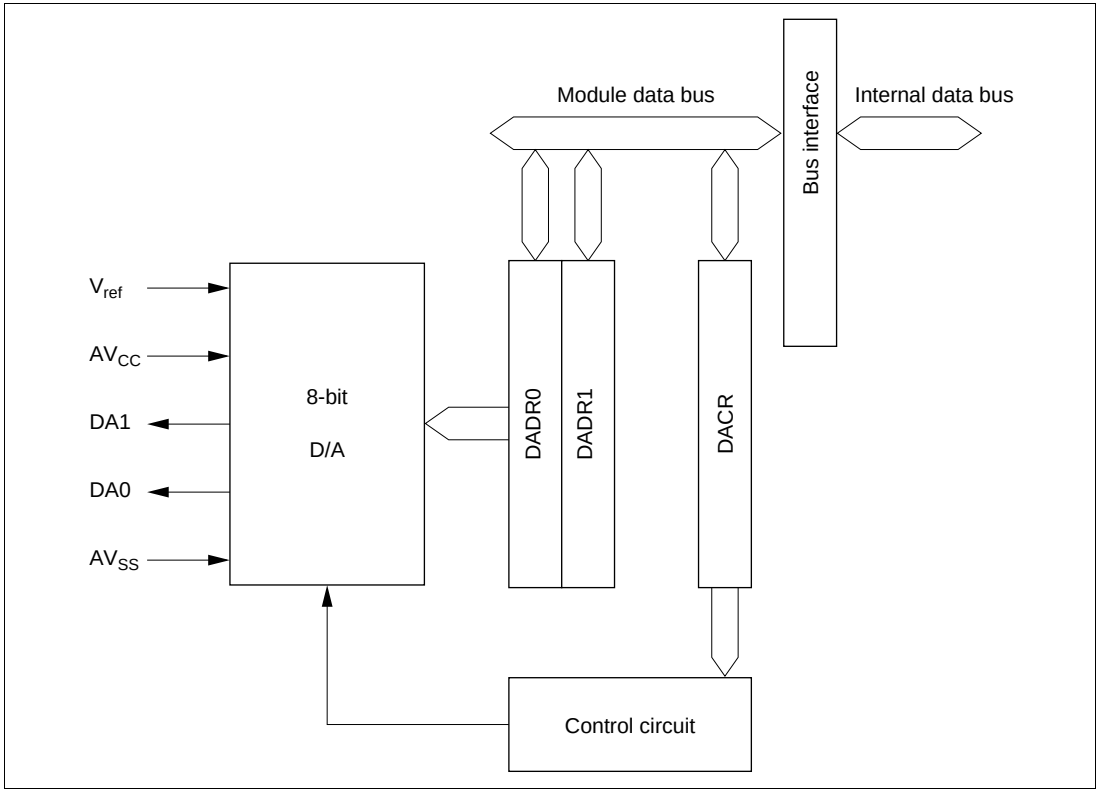



**Figure 15.1 Block Diagram of D/A Converter**

### 15.1.3 Pin Configuration

Table 15.1 summarizes the input and output pins of the D/A converter.

**Table 15.1 Pin Configuration**

| Pin Name | Symbol | I/O | Function |
|---|---|---|---|
| Analog power pin | $AV_{CC}$ | Input | Analog power source |
| Analog ground pin | $AV_{SS}$ | Input | Analog ground and reference voltage |
| Analog output pin 0 | DA0 | Output | Channel 0 analog output |
| Analog output pin 1 | DA1 | Output | Channel 1 analog output |
| Reference voltage pin | $V_{ref}$ | Input | Analog reference voltage |

### 15.1.4 Register Configuration

Table 15.2 summarizes the registers of the D/A converter.

**Table 15.2 D/A Converter Registers**

| Name | Abbreviation | R/W | Initial Value | Address* |
|---|---|---|---|---|
| D/A data register 0 | DADR0 | R/W | H'00 | H'FFA4 |
| D/A data register 1 | DADR1 | R/W | H'00 | H'FFA5 |
| D/A control register | DACR | R/W | H'1F | H'FFA6 |
| Module stop control register | MSTPCR | R/W | H'3FFF | H'FF3C |

Note:* Lower 16 bits of the address.

## 15.2 Register Descriptions

### 15.2.1 D/A Data Registers 0 and 1 (DADR0, DADR1)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

DADR0 and DADR1 are 8-bit readable/writable registers that store data for conversion.

Whenever output is enabled, the values in DADR0 and DADR1 are converted and output from the analog output pins.

DADR0 and DADR1 are each initialized to H'00 by a reset and in hardware standby mode.

### 15.2.2 D/A Control Register (DACR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | DAOE1 | DAOE0 | DAE | — | — | — | — | — |
| Initial value: | | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | — | — | — | — | — |

DACR is an 8-bit readable/writable register that controls the operation of the D/A converter.

DACR is initialized to H'1F by a reset and in hardware standby mode.

**Bit 7—D/A Output Enable 1 (DAOE1):** Controls D/A conversion and analog output for channel 1.

**Bit 7**

| DAOE1 | Description | |
|---|---|---|
| 0 | Analog output DA1 is disabled | (Initial value) |
| 1 | Channel 1 D/A conversion is enabled; analog output DA1 is enabled | |

**Bit 6—D/A Output Enable 0 (DAOE0):** Controls D/A conversion and analog output for channel 0.

| Bit 6 | |
|---|---|
| **DAOE0** | **Description** |
| 0 | Analog output DA0 is disabled (Initial value) |
| 1 | Channel 0 D/A conversion is enabled; analog output DA0 is enabled |

**Bit 5—D/A Enable (DAE):** The DAOE0 and DAOE1 bits both control D/A conversion. When the DAE bit is cleared to 0, the channel 0 and 1 D/A conversions are controlled independently. When the DAE bit is set to 1, the channel 0 and 1 D/A conversions are controlled together.

Output of resultant conversions is always controlled independently by the DAOE0 and DAOE1 bits.

| Bit 7 | Bit 6 | Bit 5 | |
|---|---|---|---|
| **DAOE1** | **DAOE0** | **DAE** | **Description** |
| 0 | 0 | * | Channel 0 and 1 D/A conversions disabled |
| | 1 | 0 | Channel 0 D/A conversion enabled<br>Channel 1 D/A conversion disabled |
| | | 1 | Channel 0 and 1 D/A conversions enabled |
| 1 | 0 | 0 | Channel 0 D/A conversion disabled<br>Channel 1 D/A conversion enabled |
| | | 1 | Channel 0 and 1 D/A conversions enabled |
| | 1 | * | Channel 0 and 1 D/A conversions enabled |

*: Don't care

If the H8S/2345 Series enters software standby mode when D/A conversion is enabled, the D/A output is held and the analog power current is the same as during D/A conversion. When it is necessary to reduce the analog power current in software standby mode, clear the DAE, DAOE0 and DAOE1 bits to 0 to disable D/A output.

**Bits 4 to 0—Reserved:** Read-only bits, always read as 1.

### 15.2.3 Module Stop Control Register (MSTPCR)

| | | MSTPCRH | | | | | | | | MSTPCRL | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | | |
| Initial value | : | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the MSTP10 bit in MSTPCR is set to 1, D/A converter operation stops at the end of the bus cycle and a transition is made to module stop mode. Registers cannot be read or written to in module stop mode. For details, see section 19.5, Module Stop Mode.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 10—Module Stop (MSTP10):** Specifies the D/A converter module stop mode.

| Bit 10 | |
|---|---|
| **MSTP10** | **Description** |
| 0 | D/A converter module stop mode cleared |
| 1 | D/A converter module stop mode set (Initial value) |

## 15.3 Operation

The D/A converter includes D/A conversion circuits for two channels, each of which can operate independently.

D/A conversion is performed continuously while enabled by DACR. If either DADR0 or DADR1 is written to, the new data is immediately converted. The conversion result is output by setting the corresponding DAOE0 or DAOE1 bit to 1.

The operation example described in this section concerns D/A conversion on channel 0. Figure 15.2 shows the timing of this operation.

[1] Write the conversion data to DADR0.

[2] Set the DAOE0 bit in DACR to 1. D/A conversion is started and the DA0 pin becomes an output pin. The conversion result is output after the conversion time has elapsed. The output value is expressed by the following formula:

$$\frac{\text{DADR contents}}{256} \times V_{ref}$$

The conversion results are output continuously until DADR0 is written to again or the DAOE0 bit is cleared to 0.

[3] If DADR0 is written to again, the new data is immediately converted. The new conversion result is output after the conversion time has elapsed.

[4] If the DAOE0 bit is cleared to 0, the DA0 pin becomes an input pin.

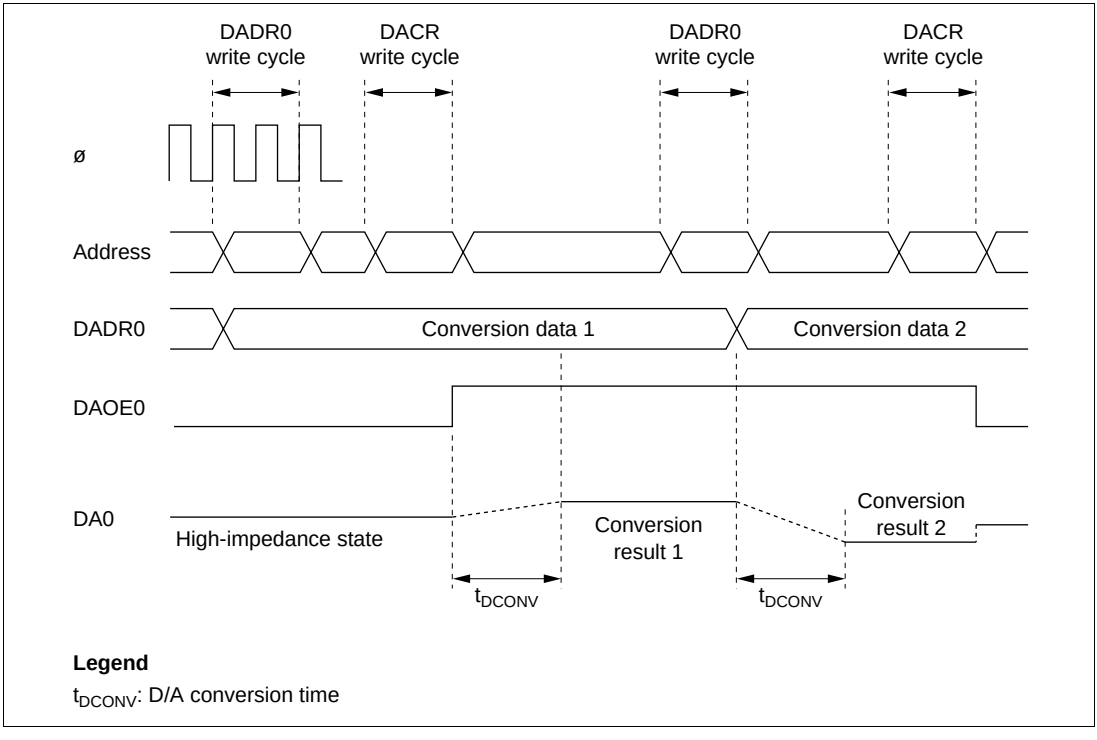



**Figure 15.2 Example of D/A Converter Operation**

## 15.4 Usage Notes

Setting range for pins other than analog power pin

(1) Relationship between $AV_{CC}$, $V_{CC}$, $AV_{SS}$, and Vss

The relationship between $AV_{CC}$, $V_{CC}$, $AV_{SS}$, and $V_{SS}$ is $AV_{CC} = V_{CC}$ and $AV_{SS} = V_{SS}$. Also, the $AV_{CC}$ and $AV_{SS}$ pins should never be left open, even if the D/A converter is not used.

(2) Vref setting range

The setting range for the reference voltage from the Vref pin is Vref ≤ $AV_{CC}$.

Note: Failure to observe (1) and (2) above could have an adverse effect on the reliability of the LSI.

# Section 16 RAM

## 16.1 Overview

The H8S/2345 and H8S/2344 have 4 kbytes of on-chip high-speed static RAM, and the H8S/2343, H8S/2341, and H8S/2340 have 2 kbytes. The RAM is connected to the CPU by a 16-bit data bus, enabling one-state access by the CPU to both byte data and word data. This makes it possible to perform fast word data transfer.

The on-chip RAM of the H8S/2345 and H8S/2344 is allocated addresses H'EC00 to H'FBFF (4 kbytes) in the normal modes (modes 1 to 3)*, and addresses H'FFEC00 to H'FFFBFF (4 kbytes) in the advanced modes (modes 4 to 7).

The on-chip RAM of the H8S/2343, H8S/2341, and H8S/2340 is allocated addresses H'F400 to H'FBFF (2 kbytes) in the normal modes (modes 1 to 3)*, and addresses H'FFF400 to H'FFFBFF (2 kbytes) in the advanced modes (modes 4 to 7).

The on-chip RAM can be enabled or disabled by means of the RAM enable bit (RAME) in the system control register (SYSCR).

Note: * ZTAT, mask ROM, and ROMless versions only.

### 16.1.1 Block Diagram

Figure 16.1 shows a block diagram of the on-chip RAM.

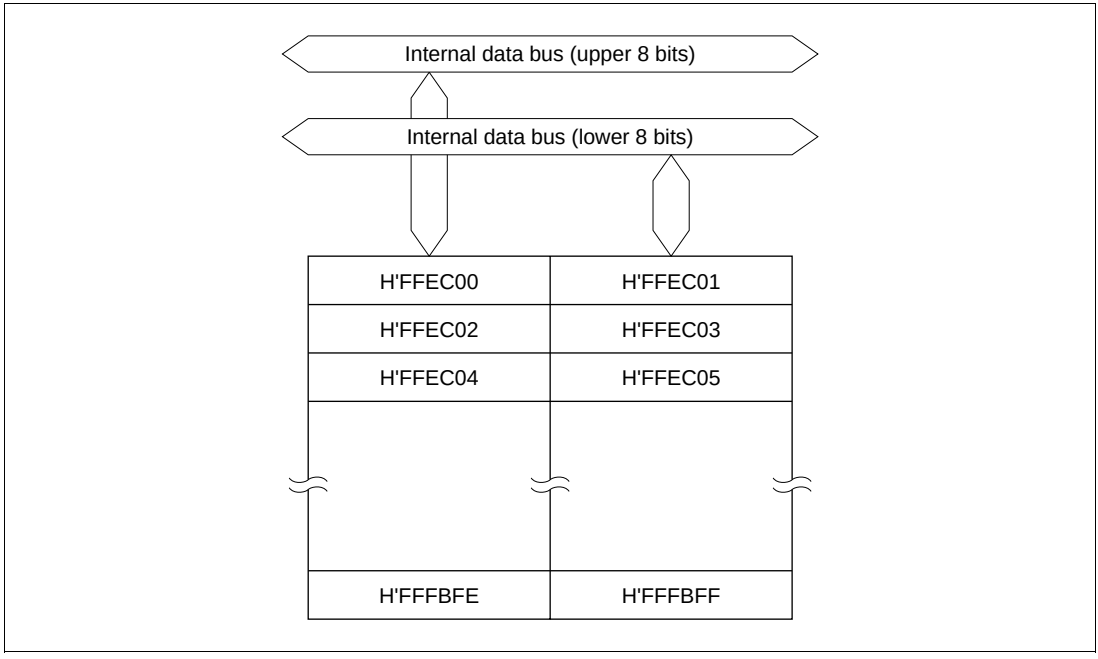



**Figure 16.1 Block Diagram of RAM (H8S/2345, Advanced Mode)**

### 16.1.2 Register Configuration

The on-chip RAM is controlled by SYSCR. Table 16.1 shows the address and initial value of SYSCR.

**Table 16.1 RAM Register**

| Name | Abbreviation | R/W | Initial Value | Address* |
|---|---|---|---|---|
| System control register | SYSCR | R/W | H'01 | H'FF39 |

Note: * Lower 16 bits of the address.

## 16.2 Register Descriptions

### 16.2.1 System Control Register (SYSCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | INTM1 | INTM0 | NMIEG | — | — | RAME |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The on-chip RAM is enabled or disabled by the RAME bit in SYSCR. For details of other bits in SYSCR, see section 3.2.2, System Control Register (SYSCR).

**Bit 0—RAM Enable (RAME):** Enables or disables the on-chip RAM. The RAME bit is initialized when the reset state is released. It is not initialized in software standby mode.

| Bit 0 | | |
|---|---|---|
| **RAME** | **Description** | |
| 0 | On-chip RAM is disabled | |
| 1 | On-chip RAM is enabled | (Initial value) |

## 16.3 Operation

When the RAME bit is set to 1, accesses to addresses H'FFEC00 to H'FFFBFF (in the case of the H8S/2345 and H8S/2344) or addresses H'FFF400 to H'FFFBFF (in the case of the H8S/2343, H8S/2341, and H8S/2340) are directed to the on-chip RAM. When the RAME bit is cleared to 0, the off-chip address space is accessed.

Since the on-chip RAM is connected to the CPU by an internal 16-bit data bus, it can be written to and read in byte or word units. Each type of access can be performed in one state.

Even addresses use the upper 8 bits, and odd addresses use the lower 8 bits. Word data must start at an even address.

## 16.4 Usage Note

DTC register information can be located in addresses H'FFF800 to H'FFFBFF. When the DTC is used, the RAME bit must not be cleared to 0.

## 17.1 Overview

The H8S/2345 has 128 kbytes of on-chip ROM (flash memory, PROM, or mask ROM); the H8S/2344 has 96 kbytes of on-chip ROM (mask ROM); the H8S/2343 has 64 kbytes of on-chip ROM (mask ROM); and the H8S/2341 has 32 kbytes of on-chip ROM (mask ROM). The ROM is connected to the H8S/2000 CPU by a 16-bit data bus. The CPU accesses both byte data and word data in one state, making possible rapid instruction fetches and high-speed processing.

The on-chip ROM is enabled or disabled by setting the mode pins ($MD_2$, $MD_1$, and $MD_0$) and bit EAE in BCRL.

The flash memory versions of the H8S/2345 Series can be erased and programmed on-board as well as with a PROM programmer.

The PROM version of the H8S/2345 Series can be programmed with a PROM programmer, by setting PROM mode.

### 17.1.1 Block Diagram

Figure 17.1 shows a block diagram of the on-chip ROM.

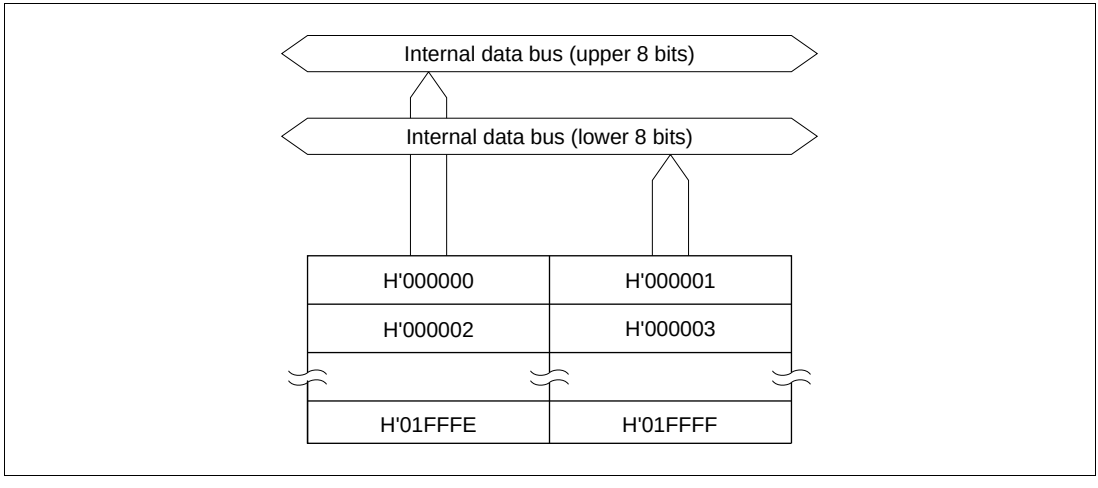



**Figure 17.1 Block Diagram of ROM (H8S/2345)**

### 17.1.2 Register Configuration

The H8S/2345's on-chip ROM is controlled by the mode pins and register BCRL. The register configuration is shown in table 17.1.

**Table 17.1 ROM Register**

| Name | Abbreviation | R/W | Initial Value | Address* |
|---|---|---|---|---|
| Mode control register | MDCR | R/W | Undefined | H'FF3B |
| Bus control register L | BCRL | R/W | Undefined | H'FED5 |

Note: * Lower 16 bits of the address.

## 17.2 Register Descriptions

### 17.2.1 Mode Control Register (MDCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | — | MDS2 | MDS1 | MDS0 |
| Initial value | : | 1 | 0 | 0 | 0 | 0 | —* | —* | —* |
| R/W | : | — | — | — | — | — | R | R | R |

Note: * Determined by pins $MD_2$ to $MD_0$.

MDCR is an 8-bit read-only register that indicates the current operating mode of the H8S/2345 Series.

**Bit 7—Reserved:** Read-only bit, always read as 1.

**Bits 6 to 3—Reserved:** Read-only bits, always read as 0.

**Bits 2 to 0—Mode Select 2 to 0 (MDS2 to MDS0):** These bits indicate the input levels at pins $MD_2$ to $MD_0$ (the current operating mode). Bits MDS2 to MDS0 correspond to pins $MD_2$ to $MD_0$. MDS2 to MDS0 are read-only bits, and cannot be written to. The mode pin ($MD_2$ to $MD_0$) input levels are latched into these bits when MDCR is read. These latches are canceled by a power-on reset, but are retained after a manual reset.

### 17.2.2 Bus Control Register L (BCRL)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | BRLE | — | EAE | — | — | — | — | WAITE |
| Initial value | : | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Enabling or disabling of part of the H8S/2345's on-chip ROM area can be selected by means of the EAE bit in BCRL. For details of the other bits in BCRL, see 6.2.5, Bus Control Register L.

**Bit 5—External Address Enable (EAE):** Selects whether addresses H'010000 to H'01FFFF are to be internal addresses or external addresses.

| Bit 5 EAE | Description |
|---|---|
| 0 | Addresses H'010000 to H'01FFFF are in on-chip ROM (H8S/2345). |
| | Addresses H'010000 to H'017FFF are in on-chip ROM and addresses H'018000 to H'01FFFF are a reserved area (in the H8S/2344). |
| | Addresses H'010000 to H'01FFFF are a reserved area (in the H8S/2343 and H8S/2341). |
| 1 | Addresses H'010000 to H'01FFFF are external addresses (external expansion mode) or a reserved area* (single-chip mode). (Initial value) |

Note: * Reserved areas should not be accessed.

## 17.3 Operation

The on-chip ROM is connected to the CPU by a 16-bit data bus, and both byte and word data can be accessed in one state. Even addresses are connected to the upper 8 bits, and odd addresses to the lower 8 bits. Word data must start at an even address.

The on-chip ROM is enabled and disabled by setting the mode pins ($MD_2$, $MD_1$, and $MD_0$) and bit EAE in BCRL. These settings are shown in tables 17.2 and 17.3.

**Table 17.2 Operating Modes and ROM Area (F-ZTAT)**

| Operating Mode | | Mode Pin | | | | BCRL | |
|---|---|---|---|---|---|---|---|
| | | FWE | $MD_2$ | $MD_1$ | $MD_0$ | EAE | On-Chip ROM |
| Mode 0 | — | 0 | 0 | 0 | 0 | — | — |
| Mode 1 | | | | | 1 | | |
| Mode 2 | | | | 1 | 0 | | |
| Mode 3 | | | | | 1 | | |
| Mode 4 | Advanced expanded mode with on-chip ROM disabled | | 1 | 0 | 0 | — | Disabled |
| Mode 5 | Advanced expanded mode with on-chip ROM disabled | | | | 1 | | |
| Mode 6 | Advanced expanded mode with on-chip ROM enabled | | | 1 | 0 | 0 | Enabled (128 kbytes)*[1] |
| | | | | | | 1 | Enabled (64 kbytes) |
| Mode 7 | Advanced single-chip mode | | | | 1 | 0 | Enabled (128 kbytes)*[1] |
| | | | | | | 1 | Enabled (64 kbytes) |
| Mode 8 | — | 1 | 0 | 0 | 0 | — | — |
| Mode 9 | | | | | 1 | | |
| Mode 10 | Boot mode (advanced expanded mode with on-chip ROM enabled)*[3] | | | 1 | 0 | 0 | Enabled (128 kbytes)*[2] |
| | | | | | | 1 | Enabled (64 kbytes) |
| Mode 11 | Boot mode (advanced single-chip mode)*[4] | | | | 1 | 0 | Enabled (128 kbytes)*[2] |
| | | | | | | 1 | Enabled (64 kbytes) |
| Mode 12 | — | | 1 | 0 | 0 | — | — |
| Mode 13 | | | | | 1 | | |
| Mode 14 | User program mode (advanced expanded mode with on-chip ROM enabled)*[3] | | | 1 | 0 | 0 | Enabled (128 kbytes)*[1] |
| | | | | | | 1 | Enabled (64 kbytes) |
| Mode 15 | User program mode (advanced single-chip mode)*[4] | | | | 1 | 0 | Enabled (128 kbytes)*[1] |
| | | | | | | 1 | Enabled (64 kbytes) |

Notes: 1. Note that in modes 6, 7, 14, and 15, the on-chip ROM that can be used after a power-on reset is the 64-kbyte area from H'000000 to H'00FFFF.

2. Note that in the mode 10 and mode 11 boot modes, the on-chip ROM that can be used immediately after all flash memory is erased by the boot program is the 64-kbyte area from H'000000 to H'00FFFF.

3. Apart from the fact that flash memory can be erased and programmed, operation is the same as in advanced expanded mode with on-chip ROM enabled.

4. Apart from the fact that flash memory can be erased and programmed, operation is the same as in advanced single-chip mode.

**Table 17.3 Operating Modes and ROM Area (ZTAT or Mask ROM)**

| | | Mode Pin | | | BCRL | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Operating Mode** | | **MD2** | **MD1** | **MD0** | **EAE** | **On-Chip ROM** | | | |
| Mode 0 | — | 0 | 0 | 0 | — | H8S/2345 | H8S/2344 | H8S/2343 | H8S/2341 |
| Mode 1 | Normal expanded mode with on-chip ROM disabled | | | 1 | | — | — | — | — |
| Mode 2*1 | Normal expanded mode with on-chip ROM enabled | | 1 | 0 | | Disabled | Disabled | Disabled | Disabled |
| Mode 3*1 | Normal single-chip mode | | | 1 | | Enabled (56 kbytes) | Enabled (56 kbytes) | Enabled (56 kbytes) | Enabled (32 kbytes) |
| Mode 4 | Advanced expanded mode with on-chip ROM disabled | 1 | 0 | 0 | — | Disabled | Disabled | Disabled | Disabled |
| Mode 5 | Advanced expanded mode with on-chip ROM disabled | | | 1 | | | | | |
| Mode 6*1 | Advanced expanded mode with on-chip ROM enabled | | 1 | 0 | 0 | Enabled (128 kbytes)*2 | Enabled*2 (96 kbytes) | Enabled (64 kbytes) | Enabled (32 kbytes) |
| | | | | | 1 | Enabled (64 kbytes) | Enabled (64 kbytes) | | |
| Mode 7*1 | Advanced single-chip mode | | | 1 | 0 | Enabled (128 kbytes)*2 | Enabled*2 (96 kbytes) | | |
| | | | | | 1 | Enabled (64 kbytes) | Enabled (64 kbytes) | | |

Notes: 1. Not used on ROMless version.

2. In H8S/2345 modes 6 and 7, the on-chip ROM available after a power-on reset is the 64-kbyte area comprising addresses H'000000 to H'00FFFF.

## 17.4 PROM Mode

### 17.4.1 PROM Mode Setting

The PROM version of the H8S/2345 suspends its microcontroller functions when placed in PROM mode, enabling the on-chip PROM to be programmed. This programming can be done with a PROM programmer set up in the same way as for the HN27C101 EPROM ($V_{PP}$ = 12.5 V). Use of a 100-pin/32-pin socket adapter enables programming with a commercial PROM programmer.

Note that the PROM programmer should not be set to page mode as the H8S/2345 does not support page programming.

Table 17.4 shows how PROM mode is selected.

**Table 17.4 Selecting PROM Mode**

| Pin Names | Setting |
|---|---|
| $MD_2$, $MD_1$, $MD_0$ | Low |
| $\overline{STBY}$ | |
| $PA_2$, $PA_1$ | High |

### 17.4.2 Socket Adapter and Memory Map

Programs can be written and verified by attaching a socket adapter to the PROM programmer to convert from a 100-pin arrangement to a 32-pin arrangement. Table 17.5 gives ordering information for the socket adapter, and figure 17.2 shows the wiring of the socket adapter. Figure 17.3 shows the memory map in PROM mode.

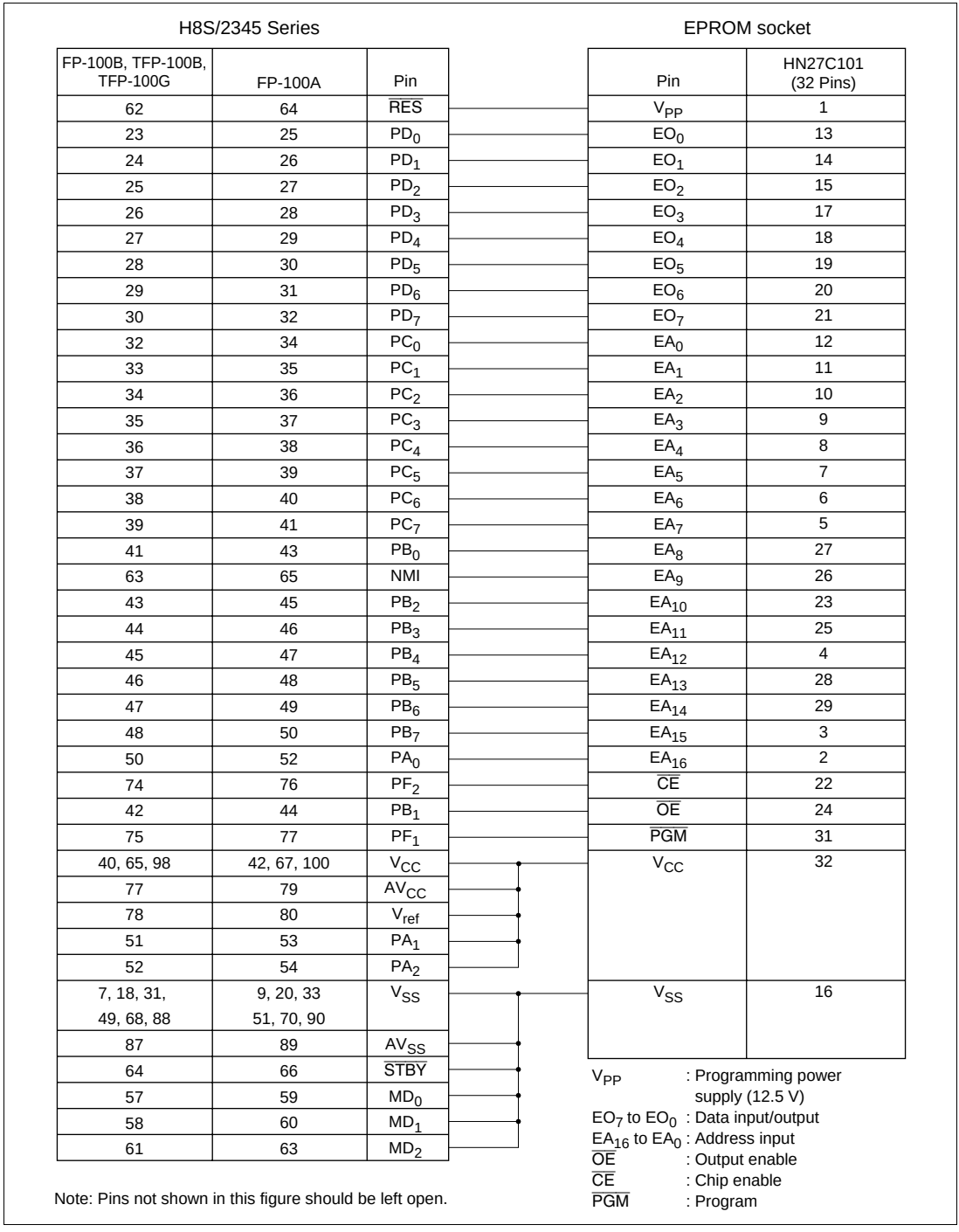



**Figure 17.2 Wiring of 100-Pin/32-Pin Socket Adapter**

**Table 17.5 Socket Adapter**

| | | Socket Adapter | |
|---|---|---|---|
| **Microcontroller** | **Package** | **MINATO ELECTRONICS INC.** | **DATA I/O CO.** |
| H8S/2345 | 100-pin TQFP (TFP-100B) | ME2345ESNS1H | H72345T100D3201 |
| | 100-pin TQFP (TFP-100G) | ME2345ESMS1H | H7234GT100D3201 |
| | 100-pin QFP (FP-100A) | ME2345ESFS1H | H7234AQ100D3201 |
| | 100-pin QFP (FP-100B) | ME2345ESHS1H | H7234BQ100D3201 |

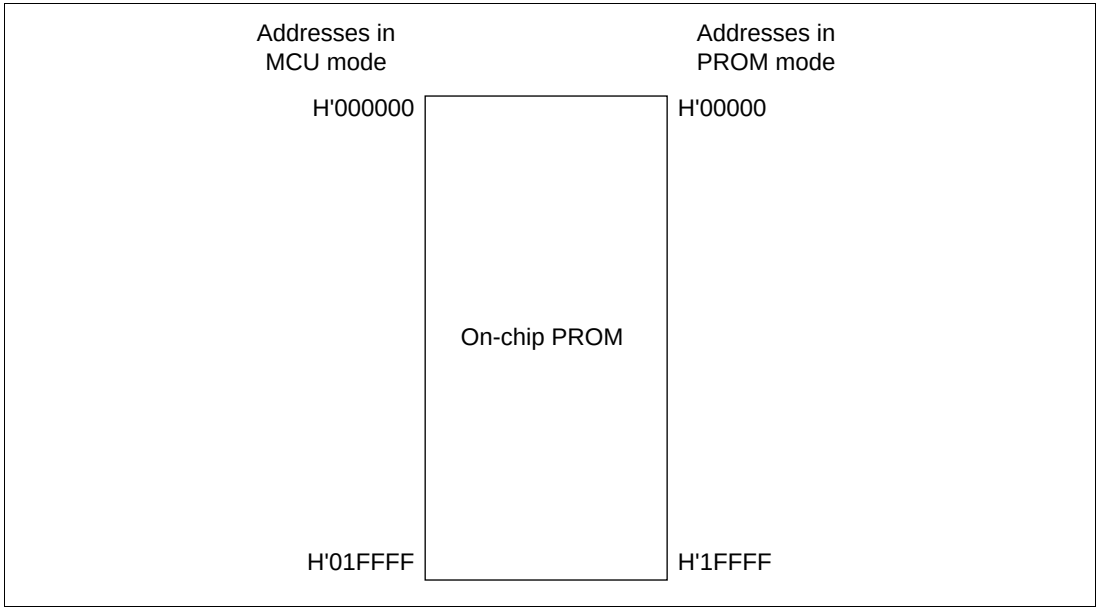



**Figure 17.3 Memory Map in PROM Mode**

## 17.5 Programming

### 17.5.1 Overview

Table 17.6 shows how to select the program, verify, and program-inhibit modes in PROM mode.

**Table 17.6 Mode Selection in PROM Mode**

| | Pins | | | | | | |
|---|---|---|---|---|---|---|---|
| **Mode** | **$\overline{CE}$** | **$\overline{OE}$** | **$\overline{PGM}$** | **$V_{PP}$** | **$V_{CC}$** | **$EO_7$ to $EO_0$** | **$EA_{16}$ to $EA_0$** |
| Program | L | H | L | $V_{PP}$ | $V_{CC}$ | Data input | Address input |
| Verify | L | L | H | $V_{PP}$ | $V_{CC}$ | Data output | Address input |
| Program-inhibit | L | L | L | $V_{PP}$ | $V_{CC}$ | High impedance | Address input |
| | L | H | H | | | | |
| | H | L | L | | | | |
| | H | H | H | | | | |

Legend

L : Low voltage level
H : High voltage level
$V_{PP}$ : $V_{PP}$ voltage level
$V_{CC}$ : $V_{CC}$ voltage level

Programming and verification should be carried out using the same specifications as for the standard HN27C101 EPROM.

However, do not set the PROM programmer to page mode, as the H8S/2345 does not support page programming. A PROM programmer that only supports page programming cannot be used. When choosing a PROM programmer, check that it supports high-speed programming in byte units. Always set addresses within the range H'00000 to H'1FFFF.

### 17.5.2 Programming and Verification

An efficient, high-speed programming procedure can be used to program and verify PROM data. This procedure writes data quickly without subjecting the chip to voltage stress or sacrificing data reliability. It leaves the data H'FF in unused addresses. Figure 17.4 shows the basic high-speed programming flowchart. Tables 17.7 and 17.8 list the electrical characteristics of the chip during programming. Figure 17.5 shows a timing chart.

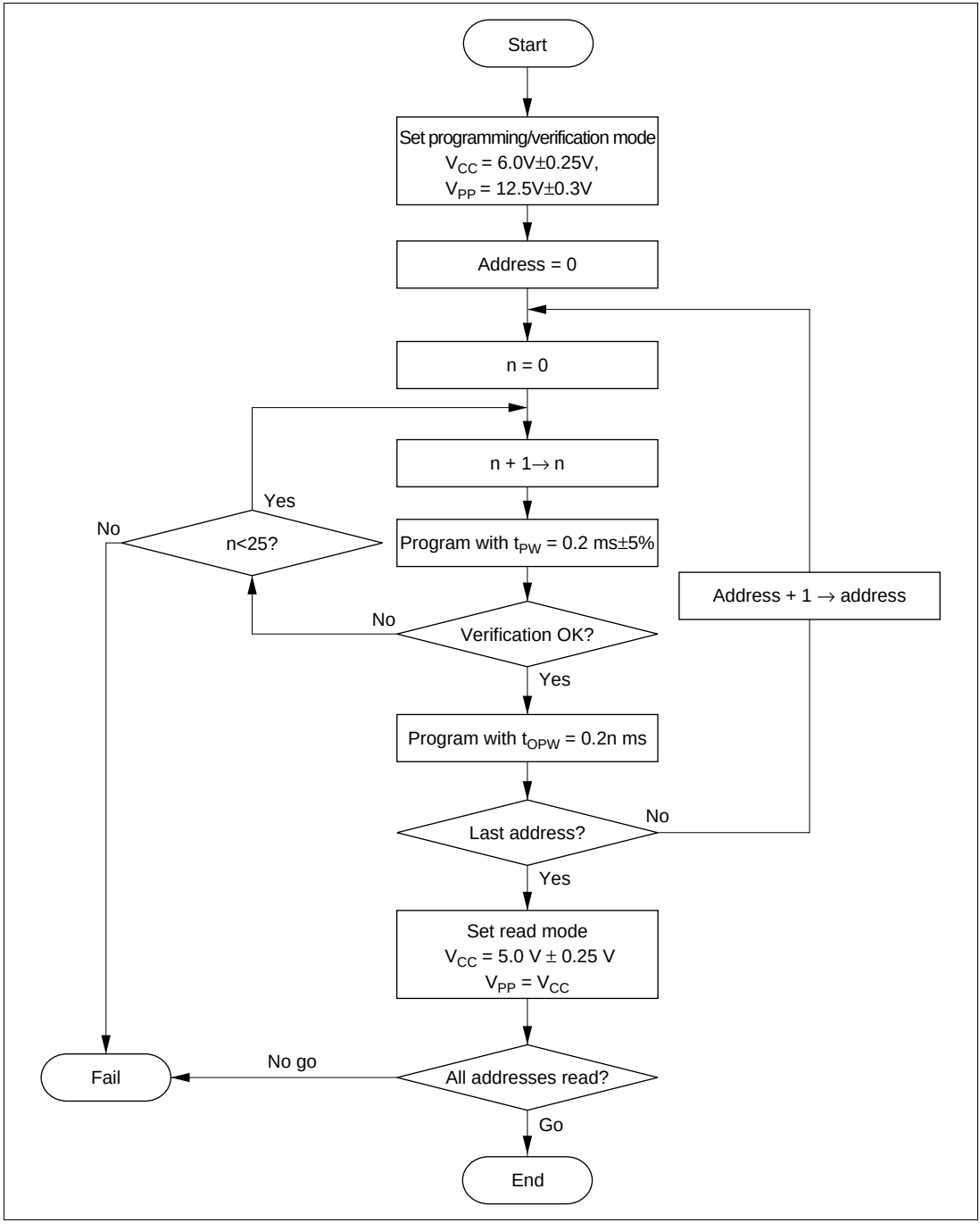



**Figure 17.4 High-Speed Programming Flowchart**

**Table 17.7 DC Characteristics in PROM Mode**
**(When $V_{CC}$ = 6.0 V ± 0.25 V, $V_{PP}$ = 12.5 V ± 0.3 V, $V_{SS}$ = 0 V, $T_a$ = 25°C ± 5°C)**

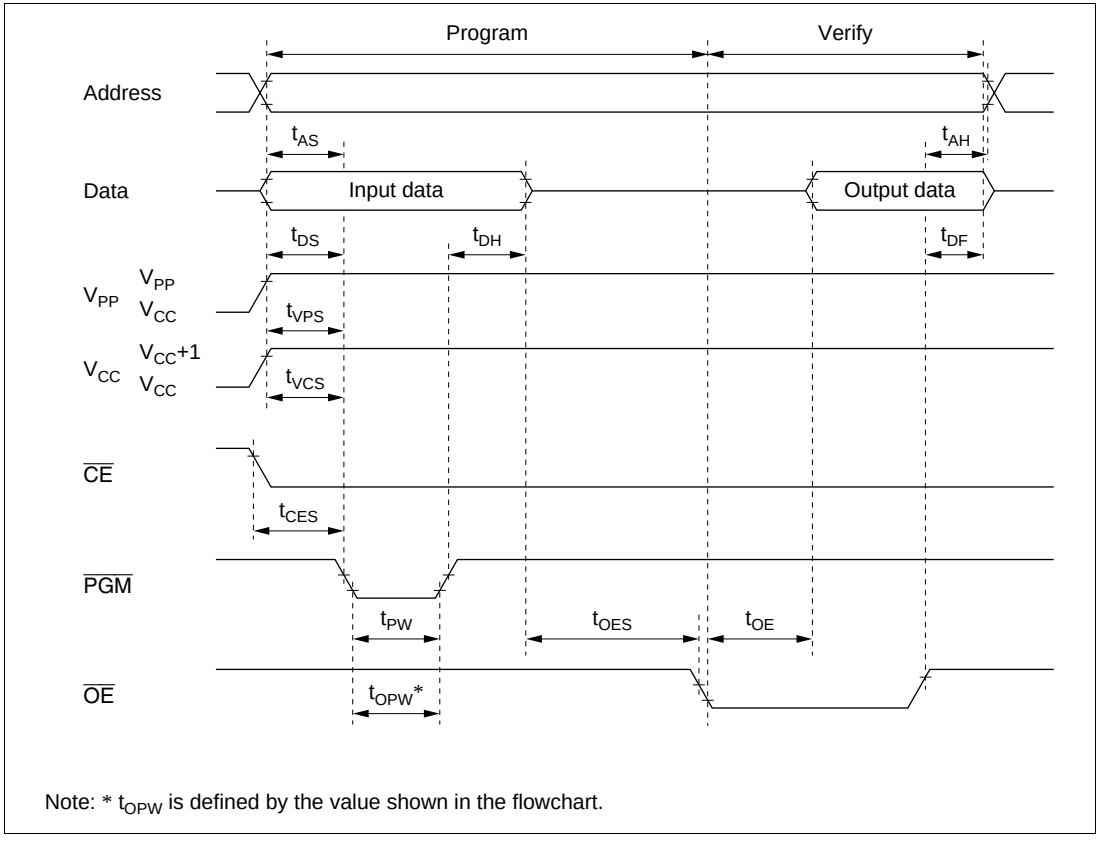
| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|
| Input high voltage | $EO_7$ to $EO_0$, $EA_{16}$ to $EA_0$, $\overline{OE}$, $\overline{CE}$, $\overline{PGM}$ | $V_{IH}$ | 2.4 | — | $V_{CC}$ + 0.3 | V | |
| Input low voltage | $EO_7$ to $EO_0$, $EA_{16}$ to $EA_0$, $\overline{OE}$, $\overline{CE}$, $\overline{PGM}$ | $V_{IL}$ | –0.3 | — | 0.8 | V | |
| Output high voltage | $EO_7$ to $EO_0$ | $V_{OH}$ | 2.4 | — | — | V | $I_{OH}$ = –200 µA |
| Output low voltage | $EO_7$ to $EO_0$ | $V_{OL}$ | — | — | 0.45 | V | $I_{OL}$ = 1.6 mA |
| Input leakage current | $EO_7$ to $EO_0$, $EA_{16}$ to $EA_0$, $\overline{OE}$, $\overline{CE}$, $\overline{PGM}$ | $\lvert I_{LI} \rvert$ | — | — | 2 | µA | $V_{in}$ = 5.25 V/0.5 V |
| $V_{CC}$ current | | $I_{CC}$ | — | — | 40 | mA | |
| $V_{PP}$ current | | $I_{PP}$ | — | — | 40 | mA | |

**Table 17.8 AC Characteristics in PROM Mode**
**(When $V_{CC}$ = 6.0 V ± 0.25 V, $V_{PP}$ = 12.5 V ± 0.3 V, $T_a$ = 25°C ± 5°C)**

| Item | Symbol | Min | Typ | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|
| Address setup time | $t_{AS}$ | 2 | — | — | µs | Figure 17.5*[1] |
| $\overline{OE}$ setup time | $t_{OES}$ | 2 | — | — | µs | |
| Data setup time | $t_{DS}$ | 2 | — | — | µs | |
| Address hold time | $t_{AH}$ | 0 | — | — | µs | |
| Data hold time | $t_{DH}$ | 2 | — | — | µs | |
| Data output disable time | $t_{DF}$*[2] | — | — | 130 | ns | |
| $V_{PP}$ setup time | $t_{VPS}$ | 2 | — | — | µs | |
| Programming pulse width | $t_{PW}$ | 0.19 | 0.20 | 0.21 | ms | |
| $\overline{PGM}$ pulse width for overwrite programming | $t_{OPW}$*[3] | 0.19 | — | 5.25 | ms | |
| $V_{CC}$ setup time | $t_{VCS}$ | 2 | — | — | µs | |
| $\overline{CE}$ setup time | $t_{CES}$ | 2 | — | — | µs | |
| Data output delay time | $t_{OE}$ | 0 | — | 150 | ns | |

Notes: 1. Input pulse level: 0.8 V to 2.2 V
Input rise time and fall time ≤ 20 ns
Timing reference levels: Input: 1.0 V, 2.0 V
Output: 0.8 V, 2.0 V

2. $t_{DF}$ is defined to be when output has reached the open state, and the output level can no longer be referenced.
3. $t_{OPW}$ is defined by the value shown in the flowchart.




Note: * $t_{OPW}$ is defined by the value shown in the flowchart.

**Figure 17.5 PROM Programming/Verification Timing**

### 17.5.3 Programming Precautions

- Program using the specified voltages and timing.

  The programming voltage ($V_{PP}$) in PROM mode is 12.5 V.

  If the PROM programmer is set to Hitachi HN27C101 specifications, $V_{PP}$ will be 12.5 V. Applied voltages in excess of the specified values can permanently destroy the MCU. Be particularly careful about the PROM programmer's overshoot characteristics.

- Before programming, check that the MCU is correctly mounted in the PROM programmer. Overcurrent damage to the MCU can result if the index marks on the PROM programmer, socket adapter, and MCU are not correctly aligned.

- Do not touch the socket adapter or MCU while programming. Touching either of these can cause contact faults and programming errors.

- The MCU cannot be programmed in page programming mode. Select the programming mode carefully.

- The size of the H8S/2345 Series PROM is 128 kbytes. Always set addresses within the range H'00000 to H'1FFFF. During programming, write H'FF to unused addresses to avoid verification errors.

### 17.5.4 Reliability of Programmed Data

An effective way to assure the data retention characteristics of the programmed chips is to bake them at 150°C, then screen them for data errors. This procedure quickly eliminates chips with PROM memory cells prone to early failure.

Figure 17.6 shows the recommended screening procedure.

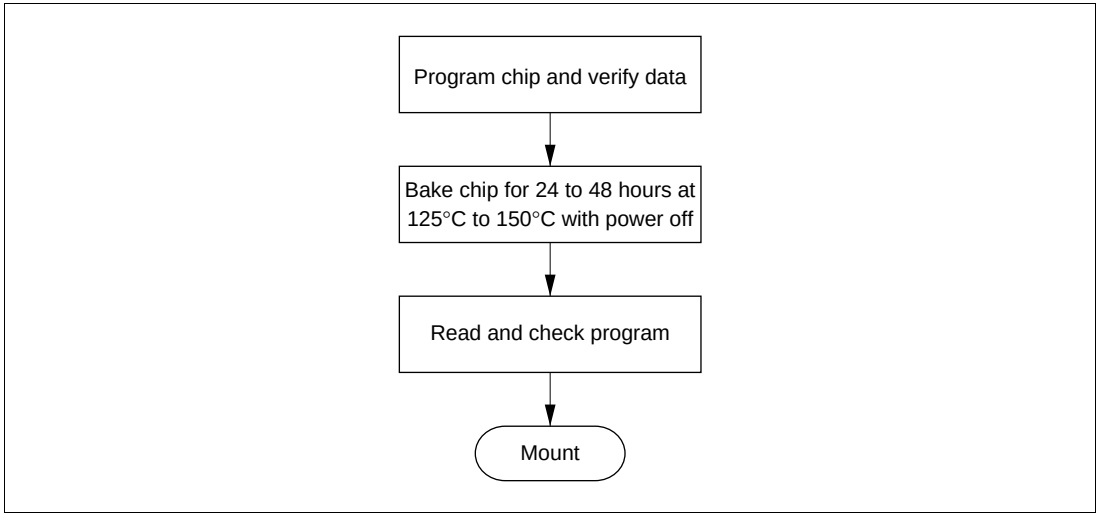



**Figure 17.6 Recommended Screening Procedure**

If a series of programming errors occurs while the same PROM programmer is being used, stop programming and check the PROM programmer and socket adapter for defects.

Please inform Hitachi of any abnormal conditions noted during or after programming or in screening of program data after high-temperature baking.

## 17.6 Overview of Flash Memory

### 17.6.1 Features

The features of the flash memory are summarized below.

- Four flash memory operating modes
  - — Program mode
  - — Erase mode
  - — Program-verify mode
  - — Erase-verify mode
- Programming/erase methods

  The flash memory is programmed 32 bytes at a time. Erasing is performed by block erase (in single-block units). When erasing multiple blocks, the individual blocks must be erased sequentially. Block erasing can be performed as required on 1-kbyte, 8-kbyte, 16-kbyte, 28-kbyte, and 32-kbyte blocks.
- Programming/erase times (5 V version)

  The flash memory programming time is 10 ms (typ.) for simultaneous 32-byte programming, equivalent to 300 μs (typ.) per byte, and the erase time is 100 ms (typ.) per block.
- Reprogramming capability

  The flash memory can be reprogrammed up to 100 times.
- On-board programming modes

  There are two modes in which flash memory can be programmed/erased/verified on-board:
  - — Boot mode
  - — User program mode
- Automatic bit rate adjustment

  With data transfer in boot mode, the bit rate of the H8S/2345 Series chip can be automatically adjusted to match the transfer bit rate of the host. (9600 bps, 4800 bps)
- Flash memory emulation by RAM

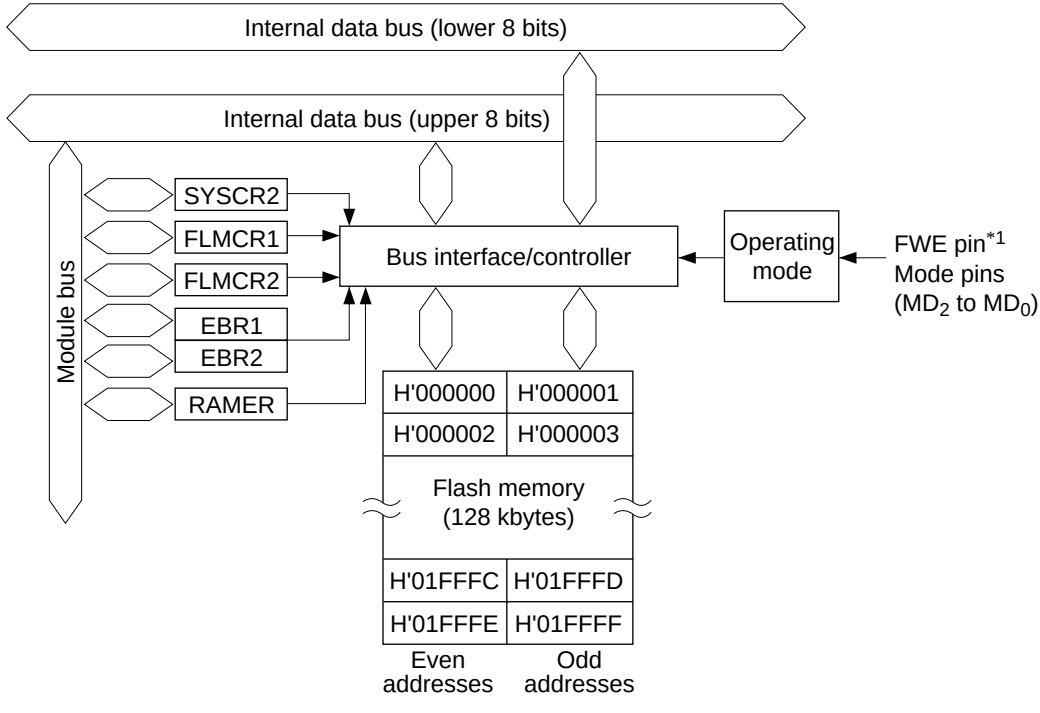
  Part of the RAM area can be overlapped onto flash memory, to emulate flash memory updates in real time.
- Protect modes

  There are three protect modes, hardware, software, and error protect, which allow protected status to be designated for flash memory program/erase/verify operations.

- Writer mode

  Flash memory can be programmed/erased in writer mode, using a PROM programmer, as well as in on-board programming mode.

### 17.6.2 Block Diagram




Legend:
SYSCR2: System control register 2*2
FLMCR1: Flash memory control register 1*2
FLMCR2: Flash memory control register 2*2
EBR1: Erase block register 1*2
EBR2: Erase block register 2*2
RAMER: RAM emulation register*2

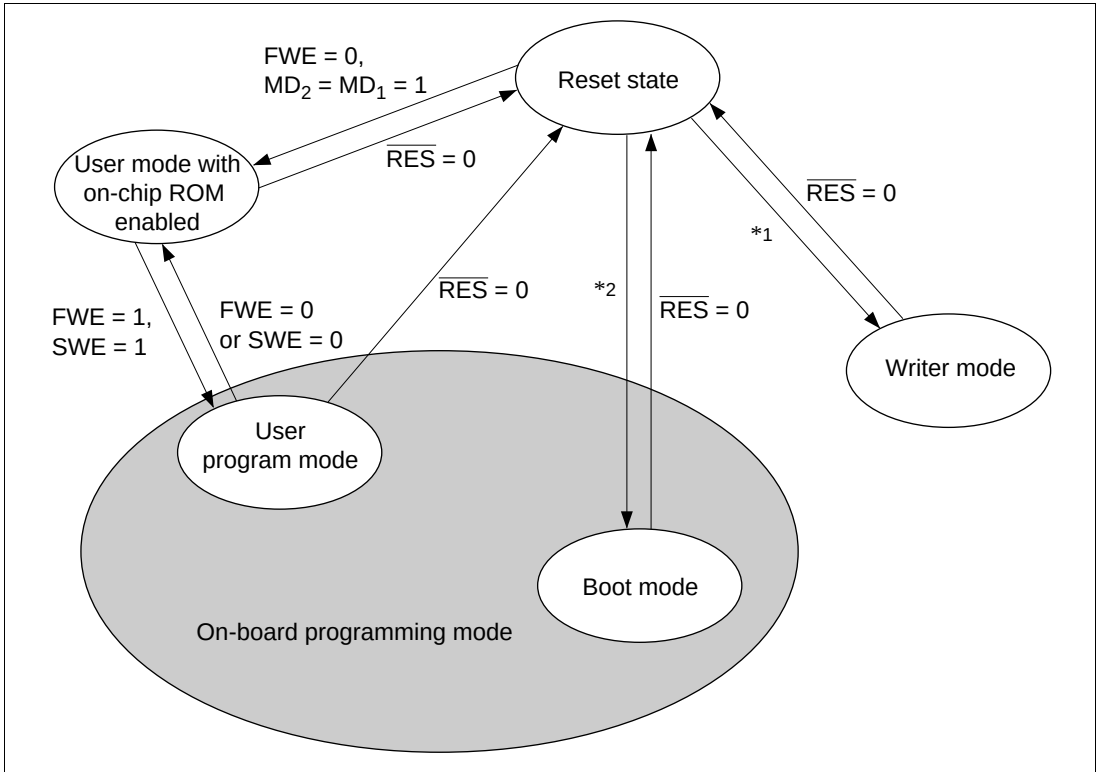
Notes: 1. Functions as FWE pin on F-ZTAT version. Functions as $\overline{\text{WDTOVF}}$ pin on ZTAT, mask ROM, and ROMless versions.
2. The flash memory control registers (SYSCR2, FLMCR1, FLMCR2, EBR1, EBR2, RAMER) are enabled on the F-ZTAT version only. They do not exist on the ZTAT, mask ROM, and ROMless versions, so an undefined value will be returned if they are read, and it is not possible to write to these registers.

**Figure 17.7 Block Diagram of Flash Memory**

### 17.6.3 Flash Memory Operating Modes

**Mode Transitions:** When the mode pins and the FWE pin are set in the reset state and a reset-start is executed, the MCU enters one of the operating modes shown in figure 17.8. In user mode, flash memory can be read but not programmed or erased.

Flash memory can be programmed and erased in boot mode, user program mode, and writer mode.



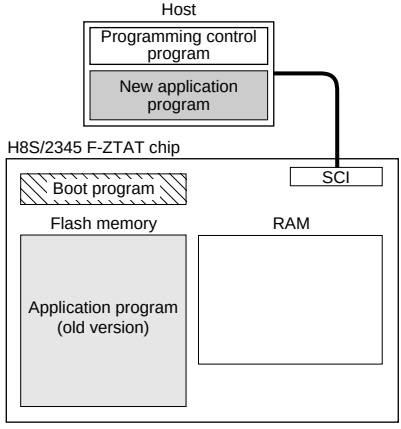

Notes: Only make a transition between user mode and user program mode when the CPU is not accessing the flash memory.

1. NMI = 1, $MD_2 = MD_1 = MD_0 = 0$, $PF_2 = 1$, $PF_1 = PF_0 = 0$
2. NMI = 1, FWE = 1, $MD_2 = 0$, $MD_1 = 1$

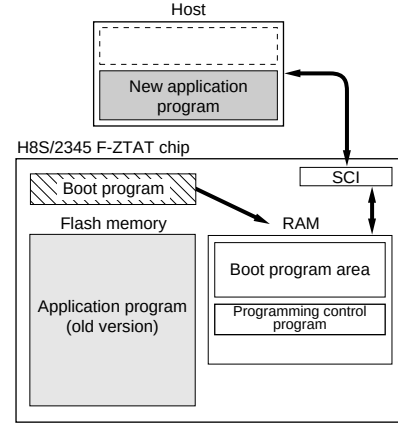
**Figure 17.8 Flash Memory Mode Transitions**

- Boot mode

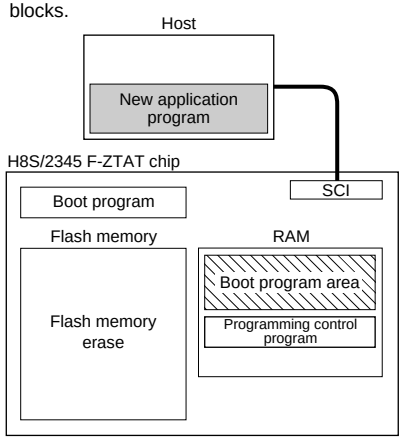
1. Initial state
   The old program version or data remains written in the flash memory. The user should prepare the programming control program and new application program beforehand in the host.




2. Programming control program transfer
   When boot mode is entered, the boot program in the H8S/2345 chip (originally incorporated in the chip) is started and the programming control program in the host is transferred to RAM via SCI communication. The boot program required for flash memory erasing is automatically transferred to the RAM boot program area.




3. Flash memory initialization
   The erase program in the boot program area (in RAM) is executed, and the flash memory is initialized (to H'FF). In boot mode, entire flash memory erasure is performed, without regard to blocks.




4. Writing new application program
   The programming control program transferred from the host to RAM is executed, and the new application program in the host is written into the flash memory.

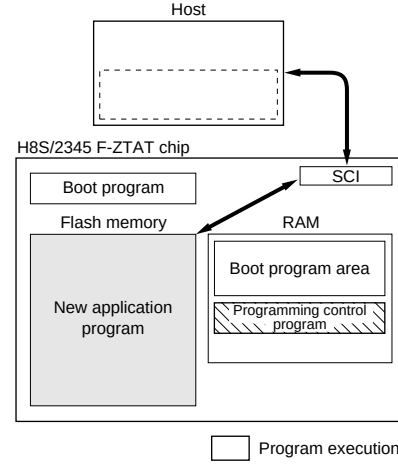
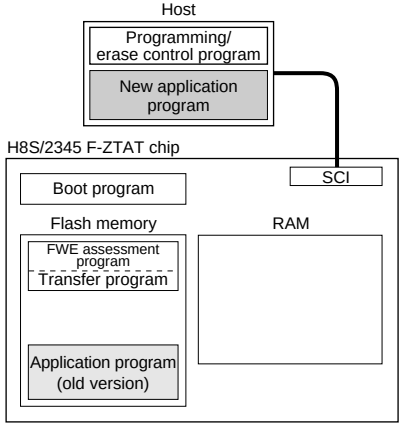


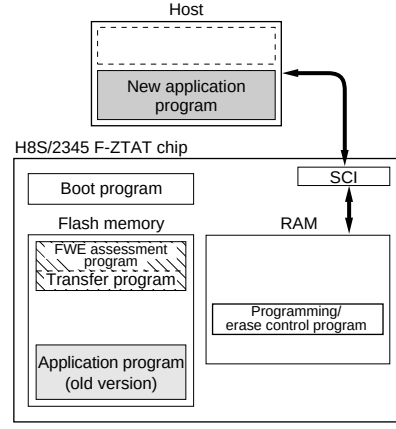

**Figure 17.9 Boot Mode**

- User program mode

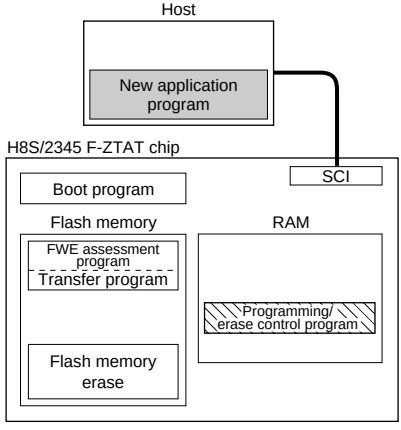
1. Initial state
   (1) The FWE assessment program that confirms that the FWE pin has been driven high, and (2) the program that will transfer the programming/ erase control program to on-chip RAM should be written into the flash memory by the user beforehand. (3) The programming/erase control program should be prepared in the host or in the flash memory.




2. Programming/erase control program transfer
   When the FWE pin is driven high, user software confirms this fact, executes the transfer program in the flash memory, and transfers the programming/erase control program to RAM.




3. Flash memory initialization
   The programming/erase program in RAM is executed, and the flash memory is initialized (to H'FF). Erasing can be performed in block units, but not in byte units.




4. Writing new application program
   Next, the new application program in the host is written into the erased flash memory blocks. Do not write to unerased blocks.

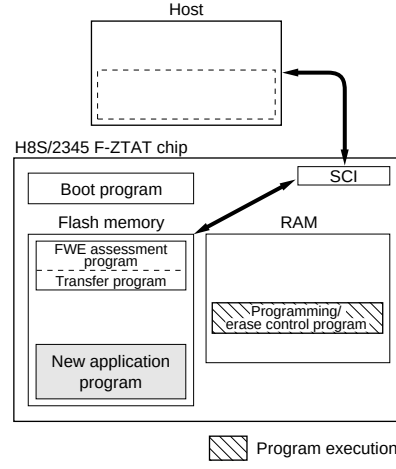



**Figure 17.10 User Program Mode (Example)**

**Flash Memory Emulation in RAM:** Emulation should be performed in user mode or user program mode. When the emulation block set in RAMER is accessed while the emulation function is being executed, data written in the overlap RAM is read.

- Reading Overlap Data in User Mode and User Program Mode

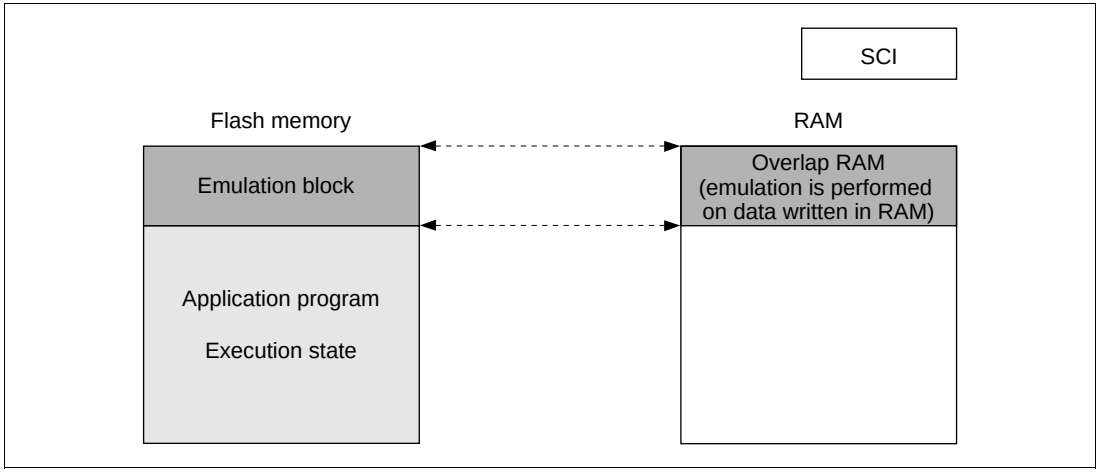



**Figure 17.11 Reading Overlap Data in User Mode and User Program Mode**

- Writing Overlap Data in User Program Mode

  When overlap RAM data is confirmed, the RAMS bit is cleared, RAM overlap is released, and writes should actually be performed to the flash memory.

  When the programming control program is transferred to RAM, ensure that the transfer destination and the overlap RAM do not overlap, as this will cause data in the overlap RAM to be rewritten.

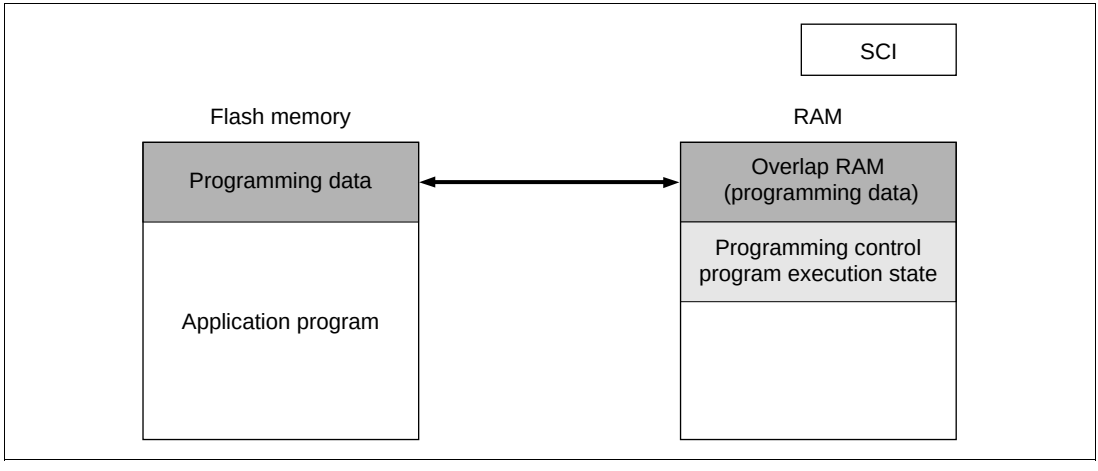



**Figure 17.12 Writing Overlap Data in User Program Mode**

**Table 17.9 Differences between Boot Mode and User Program Mode**

| | Boot Mode | User Program Mode |
|---|---|---|
| Entire memory erase | Yes | Yes |
| Block erase | No | Yes |
| Programming control program* | Program/program-verify | Program/program-verify<br>Erase/erase-verify |

Note: * To be provided by the user, in accordance with the recommended algorithm.

**Block Configuration:** The flash memory is divided into four 1-kbyte blocks, one 28-kbyte block, one 16-kbyte block, two 8-kbyte blocks, and two 32-kbyte blocks.

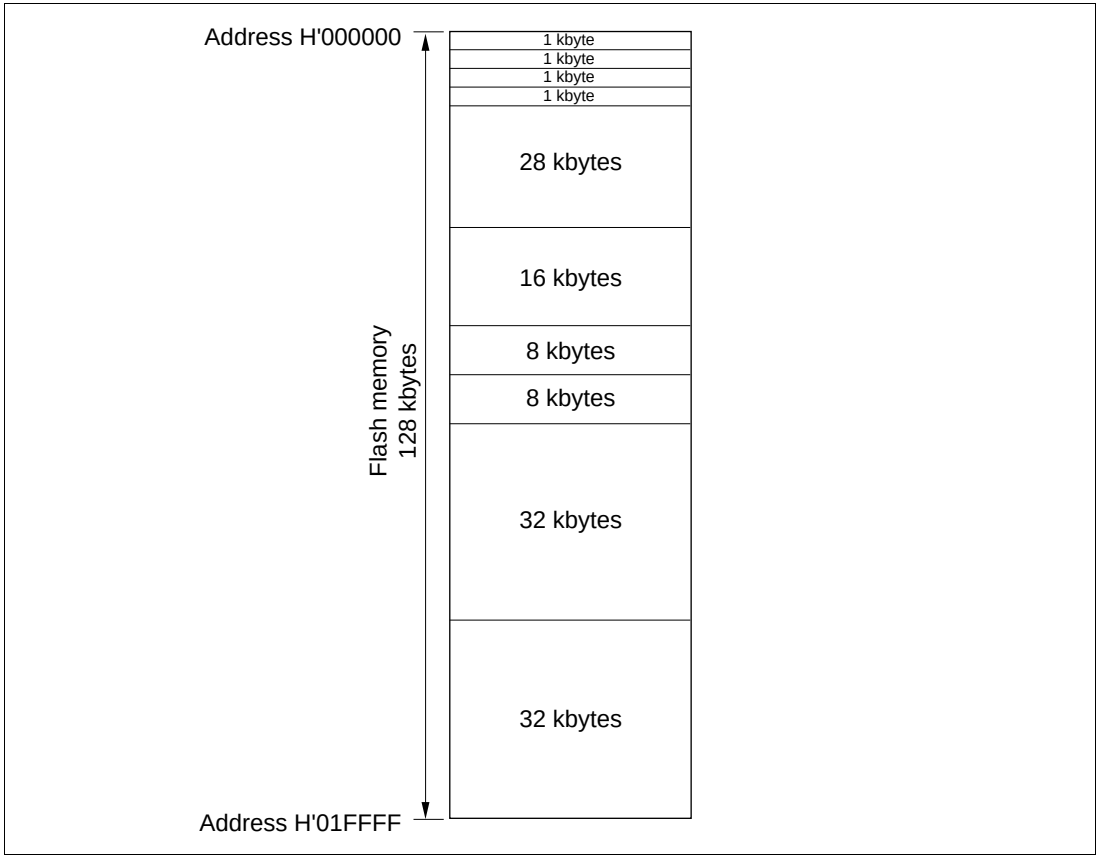



**Figure 17.13 Flash Memory Block Configuration**

### 17.6.4 Pin Configuration

The flash memory is controlled by means of the pins shown in table 17.10.

**Table 17.10 Flash Memory Pins**

| Pin Name | Abbreviation | I/O | Function |
|---|---|---|---|
| Reset | $\overline{\text{RES}}$ | Input | Reset |
| Flash write enable | FWE* | Input | Flash program/erase protection by hardware |
| Mode 2 | $MD_2$ | Input | Sets MCU operating mode |
| Mode 1 | $MD_1$ | Input | Sets MCU operating mode |
| Mode 0 | $MD_0$ | Input | Sets MCU operating mode |
| Port $F_2$ | $PF_2$ | Input | Sets MCU operating mode in writer mode |
| Port $F_1$ | $PF_1$ | Input | Sets MCU operating mode in writer mode |
| Port $F_0$ | $PF_0$ | Input | Sets MCU operating mode in writer mode |
| Transmit data | TxD1 | Output | Serial transmit data output |
| Receive data | RxD1 | Input | Serial receive data input |

Note: * FWE pin functions as $\overline{\text{WDTOVF}}$ pin on ZTAT, mask ROM, and ROMless versions.

### 17.6.5 Register Configuration

The registers used to control the on-chip flash memory when enabled are shown in table 17.11. In order for these registers to be accessed, the FLSHE bit must be set to 1 in SYSCR2.

**Table 17.11 Flash Memory Registers**

| Register Name | Abbreviation | R/W | Initial Value | Address*[1] |
|---|---|---|---|---|
| Flash memory control register 1 | FLMCR1*[6] | R/W*[3] | H'00*[4] | H'FFC8*[2] |
| Flash memory control register 2 | FLMCR2*[6] | R/W*[3] | H'00*[5] | H'FFC9*[2] |
| Erase block register 1 | EBR1*[6] | R/W*[3] | H'00*[5] | H'FFCA*[2] |
| Erase block register 2 | EBR2*[6] | R/W*[3] | H'00*[5] | H'FFCB*[2] |
| System control register 2 | SYSCR2 | R/W | H'00 | H'FF42 |
| RAM emulation register | RAMER | R/W | H'00 | H'FEDB |

Notes: 1. Lower 16 bits of the address.
2. Flash memory registers share addresses with other registers. Register selection is performed by the FLSHE bit in the system control register 2 (SYSCR2).
3. In modes in which the on-chip flash memory is disabled (modes 4 and 5), a read will return H'00, and writes are invalid. Writes are also disabled when the FWE bit is cleared to 0 in FLMCR1.
4. When a high level is input to the FWE pin, the initial value is H'80.
5. When a low level is input to the FWE pin, or if a high level is input and the SWE bit in FLMCR1 is not set, these registers are initialized to H'00.
6. FLMCR1, FLMCR2, EBR1, and EBR2 are 8-bit registers. Only byte accesses are valid for these registers, the access requiring 2 states.

The registers listed in table 7.11 are enabled on the F-ZTAT version only. They do not exist on the ZTAT, mask ROM, and ROMless versions, so an undefined value will be returned if they are read, and it is not possible to write to these registers.

## 17.7 Register Descriptions

### 17.7.1 Flash Memory Control Register 1 (FLMCR1)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FWE | SWE | — | — | EV | PV | E | P |
| Initial value | —* | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R | R/W | — | — | R/W | R/W | R/W | R/W |

Note: * Determined by the state of the FWE pin.

FLMCR1 is an 8-bit register used for flash memory operating mode control. Program-verify mode or erase-verify mode is entered by setting SWE to 1 when FWE = 1. Program mode is entered by setting SWE to 1 when FWE = 1, then setting the PSU bit in FLMCR2, and finally setting the P bit. Erase mode is entered by setting SWE to 1 when FWE = 1, then setting the ESU bit in FLMCR2, and finally setting the E bit. FLMCR1 is initialized by a reset, and in hardware standby mode and software standby mode. Its initial value is H'80 when a high level is input to the FWE pin, and H'00 when a low level is input. When on-chip flash memory is disabled (modes 4 and 5), a read will return H'00, and writes are invalid.

Writes to the SWE bit in FLMCR1 are enabled only when FWE = 1; writes to the EV and PV bits only when FWE=1 and SWE=1; writes to the E bit only when FWE = 1, SWE = 1, and ESU = 1; and writes to the P bit only when FWE = 1, SWE = 1, and PSU = 1.

**Bit 7—Flash Write Enable Bit (FWE):** Sets hardware protection against flash memory programming/erasing. See section 17.14, Flash Memory Programming and Erasing Precautions, before using this bit.

| Bit 7 FWE | Description |
|---|---|
| 0 | When a low level is input to the FWE pin (hardware-protected state) |
| 1 | When a high level is input to the FWE pin |

**Bit 6—Software Write Enable Bit (SWE):** Enables or disables flash memory programming. SWE should be set before setting bits ESU, PSU, EV, PV, E, P, and EB9 to EB0, and should not be cleared at the same time as these bits.

| Bit 6 SWE | Description | |
|---|---|---|
| 0 | Writes/erasing disabled | (Initial value) |
| 1 | Writes/erasing enabled<br>[Setting condition]<br>When FWE = 1 | |

**Bit 5 and 4—Reserved:** Read-only bits, always read as 0.

**Bit 3—Erase-Verify (EV):** Selects erase-verify mode transition or clearing. Do not set the SWE, ESU, PSU, PV, E, or P bit at the same time.

| Bit 3 EV | Description | |
|---|---|---|
| 0 | Erase-verify mode cleared | (Initial value) |
| 1 | Transition to erase-verify mode<br>[Setting condition]<br>When FWE = 1 and SWE = 1 | |

**Bit 2—Program-Verify (PV):** Selects program-verify mode transition or clearing. Do not set the SWE, ESU, PSU, EV, E, or P bit at the same time.

| Bit 2 PV | Description | |
|---|---|---|
| 0 | Program-verify mode cleared | (Initial value) |
| 1 | Transition to program-verify mode<br>[Setting condition]<br>When FWE = 1 and SWE = 1 | |

**Bit 1—Erase (E):** Selects erase mode transition or clearing. Do not set the SWE, ESU, PSU, EV, PV, or P bit at the same time.

| Bit 1<br>E | Description | |
|---|---|---|
| 0 | Erase mode cleared | (Initial value) |
| 1 | Transition to erase mode<br>[Setting condition]<br>When FWE = 1, SWE = 1, and ESU = 1 | |

**Bit 0—Program (P):** Selects program mode transition or clearing. Do not set the SWE, ESU, PSU, EV, PV, or E bit at the same time.

| Bit 0<br>P | Description | |
|---|---|---|
| 0 | Program mode cleared | (Initial value) |
| 1 | Transition to program mode<br>[Setting condition]<br>When FWE = 1, SWE = 1, and PSU = 1 | |

### 17.7.2 Flash Memory Control Register 2 (FLMCR2)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FLER | — | — | — | — | — | ESU | PSU |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R | — | — | — | — | — | R/W | R/W |

FLMCR2 is an 8-bit register that monitors the presence or absence of flash memory program/erase protection (error protection) and performs setup for flash memory program/erase mode. FLMCR2 is initialized to H'00 by a reset, and in hardware standby mode. The ESU and PSU bits are cleared to 0 in software standby mode, hardware protect mode, and software protect mode.

When on-chip flash memory is disabled, a read will return H'00.

**Bit 7—Flash Memory Error (FLER):** Indicates that an error has occurred during an operation on flash memory (programming or erasing). When FLER is set to 1, flash memory goes to the error-protect mode.

| Bit 7 FLER | Description | |
|---|---|---|
| 0 | Flash memory is operating normally | (Initial value) |
| | Flash memory program/erase protection (error protection) is disabled | |
| | [Clearing condition] | |
| | Reset or hardware standby mode | |
| 1 | An error has occurred during flash memory programming/erasing | |
| | Flash memory program/erase protection (error protection) is enabled | |
| | [Setting condition] | |
| | See section 17.10.3, Error Protection | |

**Bits 6 to 2—Reserved:** Read-only bits, always read as 0.

**Bit 1—Erase Setup (ESU):** Prepares for a transition to erase mode. Set this bit to 1 before setting the E bit to 1 in FLMCR1. Do not set the SWE, PSU, EV, PV, E, or P bit at the same time.

| Bit 1 ESU | Description | |
|---|---|---|
| 0 | Erase setup cleared | (Initial value) |
| 1 | Erase setup | |
| | [Setting condition] | |
| | When FWE = 1, and SWE = 1 | |

**Bit 0—Program Setup (PSU):** Prepares for a transition to program mode. Set this bit to 1 before setting the P bit to 1 in FLMCR1. Do not set the SWE, ESU, EV, PV, E, or P bit at the same time.

| Bit 0 PSU | Description | |
|---|---|---|
| 0 | Program setup cleared | (Initial value) |
| 1 | Program setup<br>[Setting condition]<br>When FWE = 1, and SWE = 1 | |

### 17.7.3 Erase Block Registers 1 and 2 (EBR1, EBR2)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| EBR1 | — | — | — | — | — | — | EB9 | EB8 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | — | — | — | — | R/W | R/W |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| EBR2 | EB7 | EB6 | EB5 | EB4 | EB3 | EB2 | EB1 | EB0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

EBR1 and EBR2 are registers that specify the flash memory erase area block by block; bits 1 and 2 in EBR1 and bits 7 to 0 in EBR2 are readable/writable bits. EBR1 and EBR2 are each initialized to H'00 by a reset, in hardware standby mode and software standby mode, when a low level is input to the FWE pin, and when a high level is input to the FWE pin and the SWE bit in FLMCR1 is cleared to 0. When a bit in EBR1 or EBR2 is set, the corresponding block can be erased. Other blocks are erase-protected. Blocks are erased separately (in one-block units), so set only one bit in EBR1 or EBR2 (more than one bit cannot be set to 1). To erase all blocks, erase one block at a time, once after another in sequence. Then on-chip flash memory is disabled (modes 4 and 5), a read with return H'00, and writes are disabled.

The flash memory block configuration is shown in table 17.12.

**Table 17.12 Flash Memory Erase Blocks**

| Block (Size) | Address |
|---|---|
| EB0 (1 kbyte) | H'000000 to H'0003FF |
| EB1 (1 kbyte) | H'000400 to H'0007FF |
| EB2 (1 kbyte) | H'000800 to H'000BFF |
| EB3 (1 kbyte) | H'000C00 to H'000FFF |
| EB4 (28 kbytes) | H'001000 to H'007FFF |
| EB5 (16 kbytes) | H'008000 to H'00BFFF |
| EB6 (8 kbytes) | H'00C000 to H'00DFFF |
| EB7 (8 kbytes) | H'00E000 to H'00FFFF |
| EB8 (32 kbytes) | H'010000 to H'017FFF |
| EB9 (32 kbytes) | H'018000 to H'01FFFF |

### 17.7.4 System Control Register 2 (SYSCR2)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | FLSHE | — | — | — |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | — | — | R/W | — | — | — |

SYSCR2 is an 8-bit readable/writable register that controls on-chip flash memory (in F-ZTAT versions).

SYSCR2 is initialized to H'00 by a reset and in hardware standby mode.

SYSCR2 is available only in the F-ZTAT version. In the mask ROM and ZTAT versions, this register cannot be written to and will return an undefined value if read.

**Bits 7 to 4—Reserved:** Read-only bits, always read as 0.

**Bit 3—Flash Memory Control Register Enable (FLSHE):** Controls CPU access to the flash memory control registers (FLMCR1, FLMCR2, EBR1, and EBR2). Setting the FLSHE bit to 1 enables read/write access to the flash memory control registers. If FLSHE is cleared to 0, the flash memory control registers are deselected. In this case, the flash memory control register contents are retained.

| Bit 3 FLSHE | Description | |
|---|---|---|
| 0 | Flash control registers deselected in area H'FFFFC8 to H'FFFFCB | (Initial value) |
| 1 | Flash control registers selected in area H'FFFFC8 to H'FFFFCB | |

**Bits 2 to 0—Reserved:** Read-only bits, always read as 0.

### 17.7.5 RAM Emulation Register (RAMER)

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | RAMS | RAM1 | RAM0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | — | — | — | — | — | R/W | R/W | R/W |

RAMER specifies the area of flash memory to be overlapped with part of RAM when emulating real-time flash memory programming. RAMER is initialized to H'00 by a reset and in hardware standby mode. It is not initialized in software standby mode. RAMER settings should be made in user mode or user program mode.

Flash memory area divisions are shown in table 17.13. To ensure correct operation of the emulation function, the ROM for which RAM emulation is performed should not be accessed immediately after this register has been modified. Normal execution of an access immediately after register modification is not guaranteed.

**Bits 7 to 3—Reserved:** These bits are always read as 0.

**Bit 2—RAM Select (RAMS):** Specifies selection or non-selection of flash memory emulation in RAM. When RAMS = 1, all flash memory block are program/erase-protected.

| Bit 2 RAMS | Description | |
|---|---|---|
| 0 | Emulation not selected<br>Program/erase-protection of all flash memory blocks is disabled | (Initial value) |
| 1 | Emulation selected<br>Program/erase-protection of all flash memory blocks is enabled | |

**Bits 1 and 0—Flash Memory Area Selection (RAM1, RAM0):** These bits are used together with bit 2 to select the flash memory area to be overlapped with RAM. (See table 17.13.)

**Table 17.13 Flash Memory Area Divisions**

| Addresses | Block Name | Bit 2 RAMS | Bit 1 RAM1 | Bit 0 RAM0 | Description |
|---|---|---|---|---|---|
| H'FFEC00–H'FFEFFF | RAM area 1 kbyte | 0 | * | * | RAM emulation not selected |
| H'000000–H'0003FF | EB0 (1 kbyte) | 1 | 0 | 0 | RAM emulation selected |
| H'000400–H'0007FF | EB1 (1 kbyte) | 1 | 0 | 1 | |
| H'000800–H'000BFF | EB2 (1 kbyte) | 1 | 1 | 0 | |
| H'000C00–H'000FFF | EB3 (1 kbyte) | 1 | 1 | 1 | |

*: Don't care

To use RAM for flash memory emulation, set the RAME bit of SYSCR to 1.

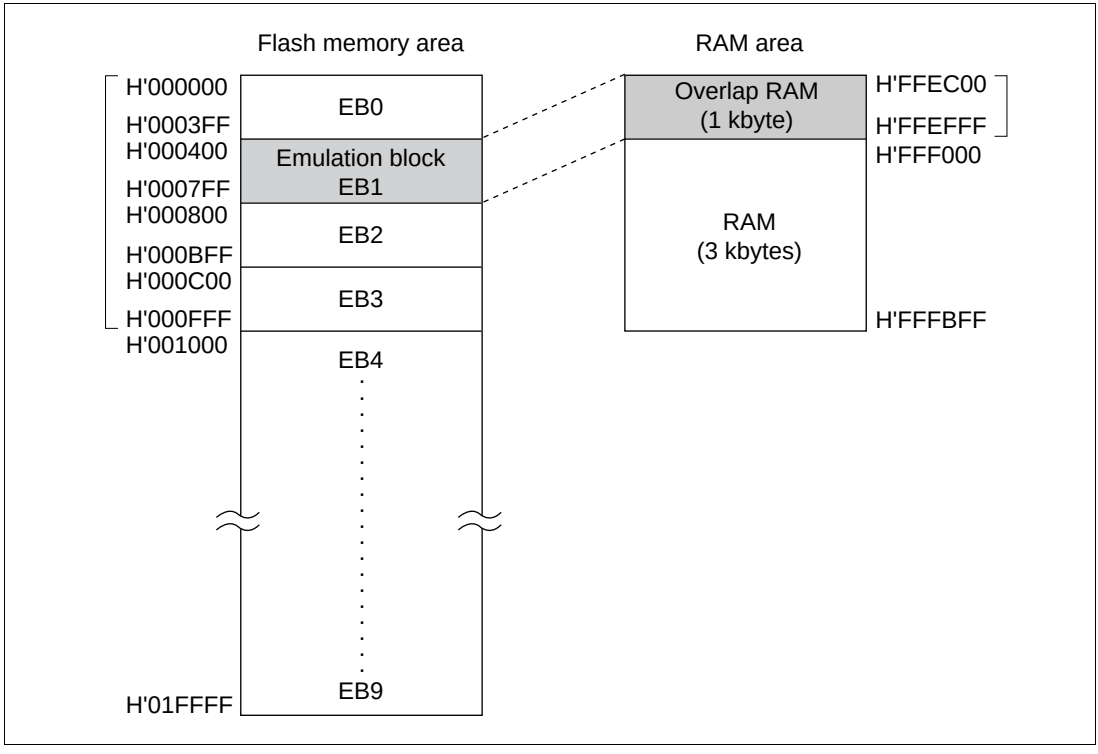



**Figure 17.14 Example of Overlap Between Flash Memory Area and RAM Area (When RAMS = 1, RAM1 = 0, and RAM0 = 1)**

## 17.8 On-Board Programming Modes

When pins are set to an on-board programming mode, they enter an on-board programming status in which program, erase, and verify operations can be performed on the on-chip flash memory. There are two on-board programming modes: boot mode and user program mode. The pin settings for transition to each of these modes are shown in table 17.14. For a diagram of the transitions to the various flash memory modes, see figure 17.8.

**Table 17.14 Setting On-Board Programming Modes**

| | Mode | | | | | |
|---|---|---|---|---|---|---|
| **Mode Name** | | **CPU Operating Mode** | **FWE** | **$MD_2$** | **$MD_1$** | **$MD_0$** |
| Boot mode | Mode 10 | Advanced expanded mode with on-chip ROM enabled | 1 | 0 | 1 | 0 |
| | Mode 11 | Advanced single-chip mode | | | | 1 |
| User program mode*[1] | Mode 14 | Advanced expanded mode with on-chip ROM enabled | 1*[2] | 1 | 1 | 0 |
| | Mode 15 | Advanced single-chip mode | | | | 1 |

Notes: 1. Normally, user mode (modes 6 and 7) should be used. Set FWE to 1 to make a transition to user program mode (modes 14 and 15) before performing a program/erase/verify operation.

2. Refer to “17.4, Notes on Programming and Erasing Flash Memory” for information on programming and clearing FWE.

### 17.8.1 Boot Mode

When boot mode is used, the flash memory programming control program must be prepared in the host beforehand. The channel 1 SCI to be used is set to asynchronous mode.

When a reset-start is executed after the H8S/2345 MCU's pins have been set to boot mode, the boot program built into the MCU is started and the programming control program prepared in the host is serially transmitted to the MCU via the SCI. In the MCU, the programming control program received via the SCI is written into the programming control program area in on-chip RAM. After the transfer is completed, control branches to the start address of the programming control program area and the programming control program execution state is entered (flash memory programming is performed).

The transferred programming control program must therefore include coding that follows the programming algorithm given later.

The system configuration in boot mode is shown in figure 17.15, and the boot program mode execution procedure in figure 17.16.

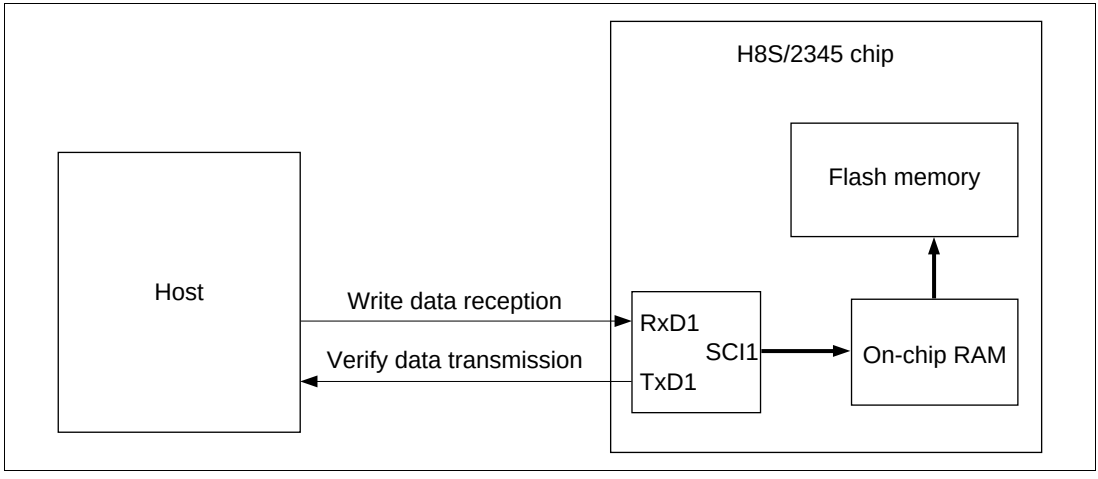
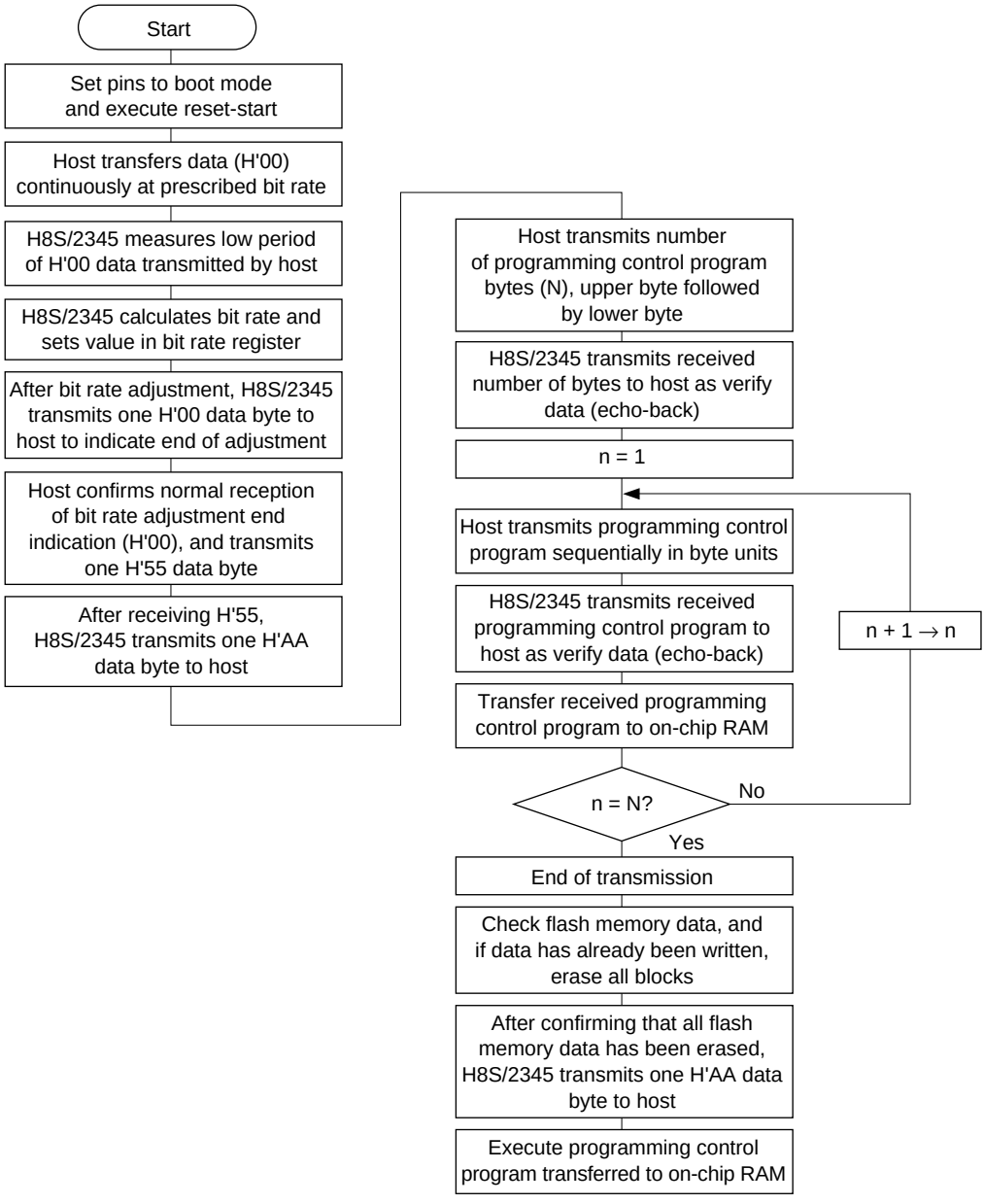



**Figure 17.15 System Configuration in Boot Mode**




Note: If a memory cell does not operate normally and cannot be erased, one H'FF byte is transmitted as an erase error, and the erase operation and subsequent operations are halted.

**Figure 17.16 Boot Mode Execution Procedure**

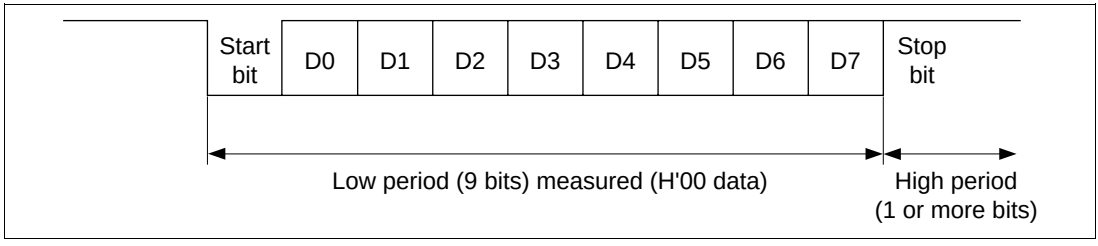



**Figure 17.17 Measurement of Low Period of Host Transmission Data**

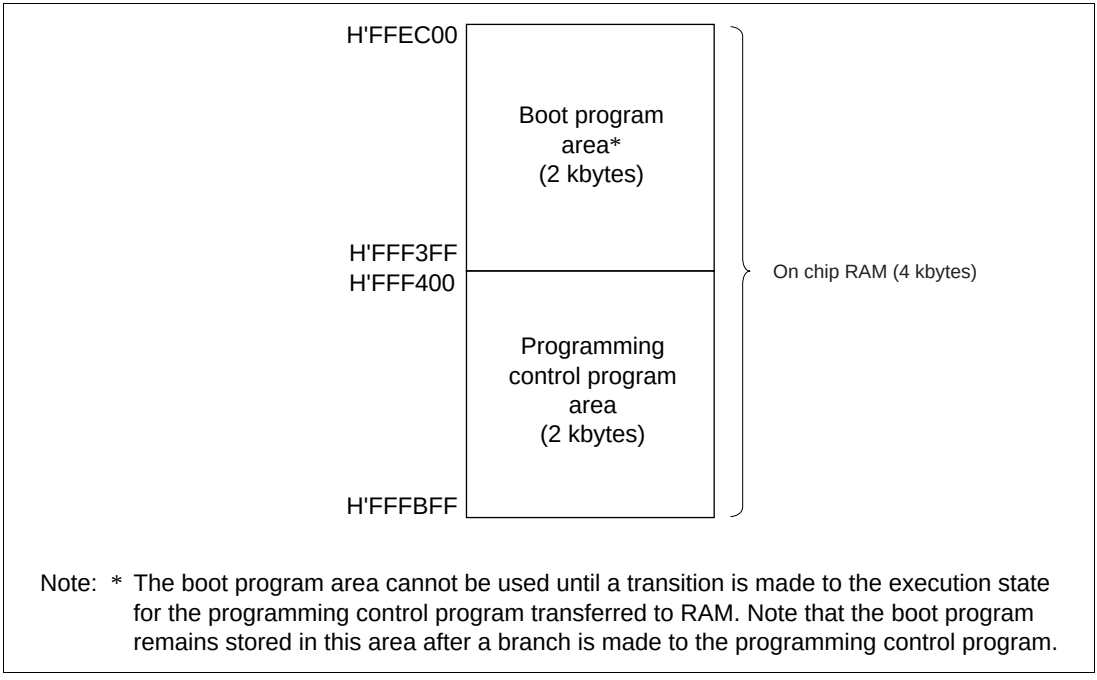
When boot mode is initiated, the H8S/2345 MCU measures the low period of the asynchronous SCI communication data (H'00) transmitted continuously from the host, see figure 17.17. The SCI transmit/receive format should be set as follows: 8-bit data, 1 stop bit, no parity. The MCU calculates the bit rate of the transmission from the host from the measured low period, and transmits one H'00 byte to the host to indicate the end of bit rate adjustment. The host should confirm that this adjustment end indication (H'00) has been received normally, and transmit one H'55 byte to the MCU. If reception cannot be performed normally, initiate boot mode again (reset), and repeat the above operations. Depending on the host's transmission bit rate and the MCU's system clock frequency, there will be a discrepancy between the bit rates of the host and the MCU. To ensure correct SCI operation, the host's transfer bit rate should be set to (4800, or 9600) bps.

Table 17.15 shows typical host transfer bit rates and system clock frequencies for which automatic adjustment of the MCU's bit rate is possible. The boot program should be executed within this system clock range.

**Table 17.15 System Clock Frequencies for which Automatic Adjustment of H8S/2345 Bit Rate is Possible**

| Host Bit Rate | System Clock Frequency for which Automatic Adjustment of H8S/2345 Bit Rate is Possible |
|---|---|
| 9600 bps | 8 MHz to 20 MHz |
| 4800 bps | 4 MHz to 20 MHz |

**On-Chip RAM Area Divisions in Boot Mode:** In boot mode, the 2 kbytes area from H'FFEC00 to H'FFF3FF is reserved for use by the boot program, as shown in figure 17.18. The area to which the programming control program is transferred is H'FFF400 to H'FFFBFF. The boot program area can be used when the programming control program transferred into RAM enters the execution state. A stack area should be set up as required.




Note: * The boot program area cannot be used until a transition is made to the execution state for the programming control program transferred to RAM. Note that the boot program remains stored in this area after a branch is made to the programming control program.

**Figure 17.18 RAM Areas in Boot Mode**

**Notes on Use of User Mode:**

- When the chip comes out of reset in boot mode, it measures the low-level period of the input at the SCI's RxD1 pin. The reset should end with RxD1 high. After the reset ends, it takes approximately 100 states before the chip is ready to measure the low-level period of the RxD1 pin.
- In boot mode, if any data has been programmed into the flash memory (if all data is not 1), all flash memory blocks are erased. Boot mode is for use when user program mode is unavailable, such as the first time on-board programming is performed, or if the program activated in user program mode is accidentally erased.
- Interrupts cannot be used while the flash memory is being programmed or erased.
- The RxD1 and TxD1 pins should be pulled up on the board.
- Before branching to the programming control program (RAM area H'FFF400), the chip terminates transmit and receive operations by the on-chip SCI (channel 1) (by clearing the RE and TE bits in SCR to 0), but the adjusted bit rate value remains set in BRR. The transmit data output pin, TxD1, goes to the high-level output state (P31DDR = 1, P31DR = 1).

The contents of the CPU's internal general registers are undefined at this time, so these registers must be initialized immediately after branching to the programming control program. In particular, since the stack pointer (SP) is used implicitly in subroutine calls, etc., a stack area must be specified for use by the programming control program.

Initial settings must also be made for the other on-chip registers.

- Boot mode can be entered by making the settings to the FWE pin and the mode pins ($MD_2$ to $MD_0$) shown in Table 17.14 and executing a reset-start. (See figure 17.37.)

  To change from boot mode to another mode (user mode, etc.), the microcomputer's internal boot mode status must first be cleared by inputting a reset using the $\overline{RES}$ pin*[1]. In this case, the $\overline{RES}$ pin must be kept low ($t_{RESW}$) for at least 20 states. (See figure 17.38.)

- Do not change the FWE pin and mode pin input levels in boot mode, and do not drive the FWE pin low while the boot program is being executed or while flash memory is being programmed or erased.*[2]

- If the FWE pin or mode pin input levels are changed (for example, from low to high) during a reset, the state of ports with multiplexed address functions and bus control output pins ($\overline{AS}$, $\overline{RD}$, $\overline{HWR}$, $\overline{LWR}$) will change according to the change in the microcomputer's operating mode*[3].

  Therefore, care must be taken to make pin settings to prevent these pins from becoming output signal pins during a reset, or to prevent collision with signals outside the microcomputer.

Notes: 1. FWE pin and mode pin input must satisfy the mode programming setup time ($t_{MDS}$) with respect to the reset release timing, as shown in figures 17.36 to 17.38.
2. For further information on FWE application and disconnection, see section 17.14, Flash Memory Programming and Erasing Precautions.
3. See appendix D, Pin States.

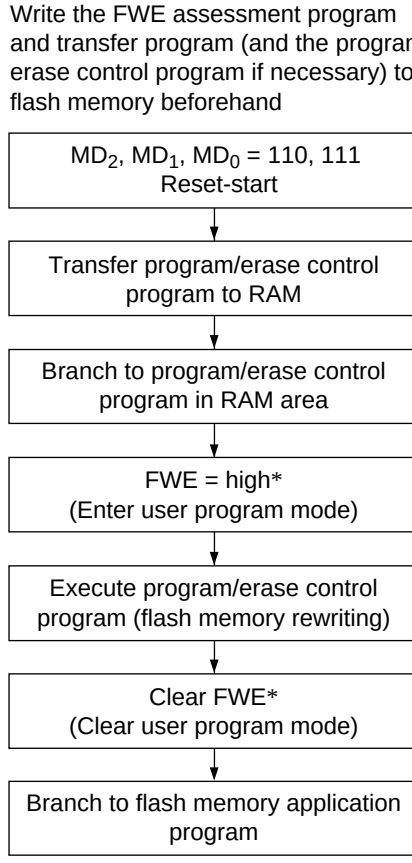
### 17.8.2 User Program Mode

When set to user program mode, the chip can program and erase its flash memory by executing a user program/erase control program. Therefore, on-board reprogramming of the on-chip flash memory can be carried out by providing on-board means of FWE control and supply of programming data, and storing a program/erase control program in part of the program area as necessary.

To select user program mode, select a mode that enables the on-chip flash memory (mode 6 or 7), and apply a high level to the FWE pin. In this mode, on-chip supporting modules other than flash memory operate as they normally would in modes 6 and 7, see figures 17.37 and 17.38.

The flash memory itself cannot be read while the SWE bit is set to 1 to perform programming or erasing, so the control program that performs programming and erasing should be run in on-chip RAM or external memory.

Figure 17.19 shows the procedure for executing the program/erase control program when transferred to on-chip RAM.




Note: Do not apply a constant high level to the FWE pin. Apply a high level to the FWE pin only when the flash memory is programmed or erased. Also, while a high level is applied to the FWE pin, the watchdog timer should be activated to prevent overprogramming or overerasing due to program runaway, etc.

* For further information on FWE application and disconnection, see section 17.14, Flash Memory Programming and Erasing Precautions.

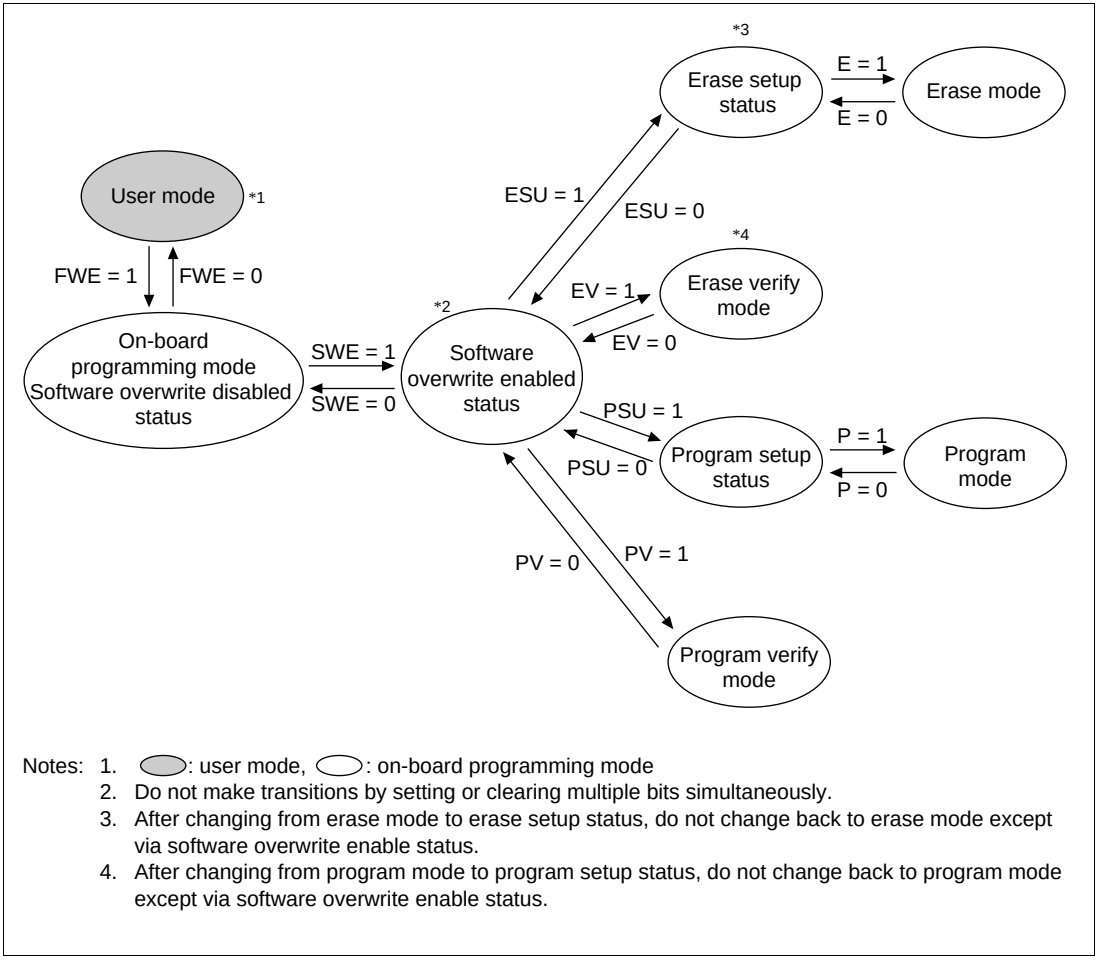
**Figure 17.19 User Program Mode Execution Procedure**

## 17.9 Programming/Erasing Flash Memory

In the on-board programming modes, flash memory programming and erasing is performed by software, using the CPU. There are four flash memory operating modes: program mode, erase mode, program-verify mode, and erase-verify mode. Transitions to these modes can be made by setting the PSU and ESU bits in FLMCR2, and the P, E, PV, and EV bits in FLMCR1. Refer to figure 17.20 regarding mode transitions using the settings of the bits in FLMCR1 and FLMCR2.

The flash memory cannot be read while being programmed or erased. Therefore, the program that controls flash memory programming/erasing (the programming control program) should be located and executed in on-chip RAM or external memory.

Notes: 1. Operation is not guaranteed if setting/resetting of the SWE, EV, PV, E, and P bits in FLMCR1, and the ESU and PSU bits in FLMCR2, is executed by a program in flash memory.
2. When programming or erasing, set FWE to 1 (programming/erasing will not be executed if FWE = 0).
3. Perform programming in the erased state. Do not perform additional programming on previously programmed addresses.




Notes: 1. ⬭: user mode, ⬭: on-board programming mode
2. Do not make transitions by setting or clearing multiple bits simultaneously.
3. After changing from erase mode to erase setup status, do not change back to erase mode except via software overwrite enable status.
4. After changing from program mode to program setup status, do not change back to program mode except via software overwrite enable status.

**Figure 17.20 Mode Transitions Using Settings of Bits in FLMCR1 and FLMCR2**

### 17.9.1 Program Mode

Follow the procedure shown in the program/program-verify flowchart in figure 17.21 to write data or programs to flash memory. Performing program operations according to this flowchart will enable data or programs to be written to flash memory without subjecting the device to voltage stress or sacrificing program data reliability. Programming should be carried out 32 bytes at a time.

The wait times (x, y, z, α, ß, γ, ε, η) after bits are set or cleared in flash memory control registers 1 and 2 (FLMCR1, FLMCR2) and the maximum number of programming operations (N) are shown in table 22.10 in section 20.1.6, Flash Memory Characteristics.

Following the elapse of (x) µs or more after the SWE bit is set to 1 in FLMCR1, 32-byte program data is stored in the program data area and reprogram data area, and the 32-byte data in the

reprogram data area written consecutively to the write addresses. The lower 8 bits of the first address written to must be H'00, H'20, H'40, H'60, H'80, H'A0, H'C0, or H'E0. Thirty-two consecutive byte data transfers are performed. The program address and program data are latched in the flash memory. A 32-byte data transfer must be performed even if writing fewer than 32 bytes; in this case, H'FF data must be written to the extra addresses.

Next, the watchdog timer is set to prevent overprogramming in the event of program runaway, etc. Set a value greater than (y + z + α + ß) μs as the WDT overflow period. After this, preparation for program mode (program setup) is carried out by setting the PSU bit in FLMCR2, and after the elapse of (y) μs or more, the operating mode is switched to program mode by setting the P bit in FLMCR1. The time during which the P bit is set is the flash memory programming time. Make a program setting so that the time for one programming operation is within the range of (z) μs.

### 17.9.2 Program-Verify Mode

In program-verify mode, the data written in program mode is read to check whether it has been correctly written in the flash memory.

After the elapse of a given programming time, the programming mode is exited (the P bit in FLMCR1 is cleared to 0, then the PSU bit in FLMCR2 is cleared to 0 at least (α) μs later). Next, the watchdog timer is cleared after the elapse of (y + z + α + β) μs or more, and the operating mode is switched to program-verify mode by setting the PV bit in FLMCR1. Before reading in program-verify mode, a dummy write of H'FF data should be made to the addresses to be read. The dummy write should be executed after the elapse of (γ) μs or more. When the flash memory is read in this state (verify data is read in 16-bit units), the data at the latched address is read. Wait at least (ε) μs after the dummy write before performing this read operation. Next, the originally written data is compared with the verify data, and reprogram data is computed (see figure 17.21) and transferred to the reprogram data area. After 32 bytes of data have been verified, exit program-verify mode, wait for at least (η) μs, then clear the SWE bit in FLMCR1 to 0. If reprogramming is necessary, set program mode again, and repeat the program/program-verify sequence as before. However, ensure that the program/program-verify sequence is not repeated more than (N) times on the same bits.

Note: An area in RAM for storing write data (32 bytes) and an area for storing rewrite data (32 bytes) are required.

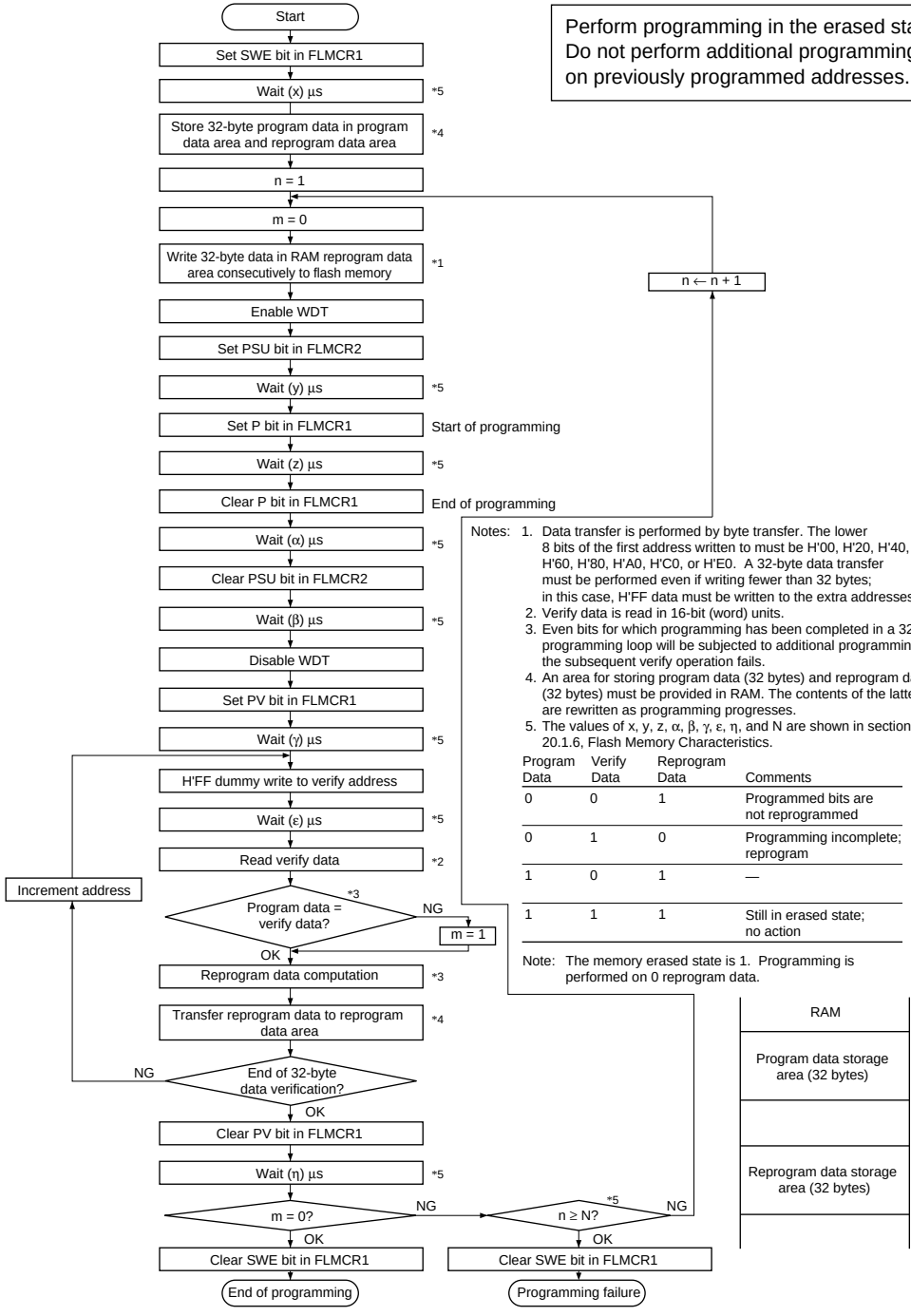



**Figure 17.21 Program/Program-Verify Flowchart**

### 17.9.3 Erase Mode

Flash memory erasing should be performed block by block following the procedure shown in the erase/erase-verify flowchart (single-block erase) shown in figure 17.22.

The wait times (x, y, z, α, ß, γ, ε, η) after bits are set or cleared in flash memory control registers 1 and 2 (FLMCR1, FLMCR2) and the maximum number of programming operations (N) are shown in table 20.10 in section 20.1.6, Flash Memory Characteristics.

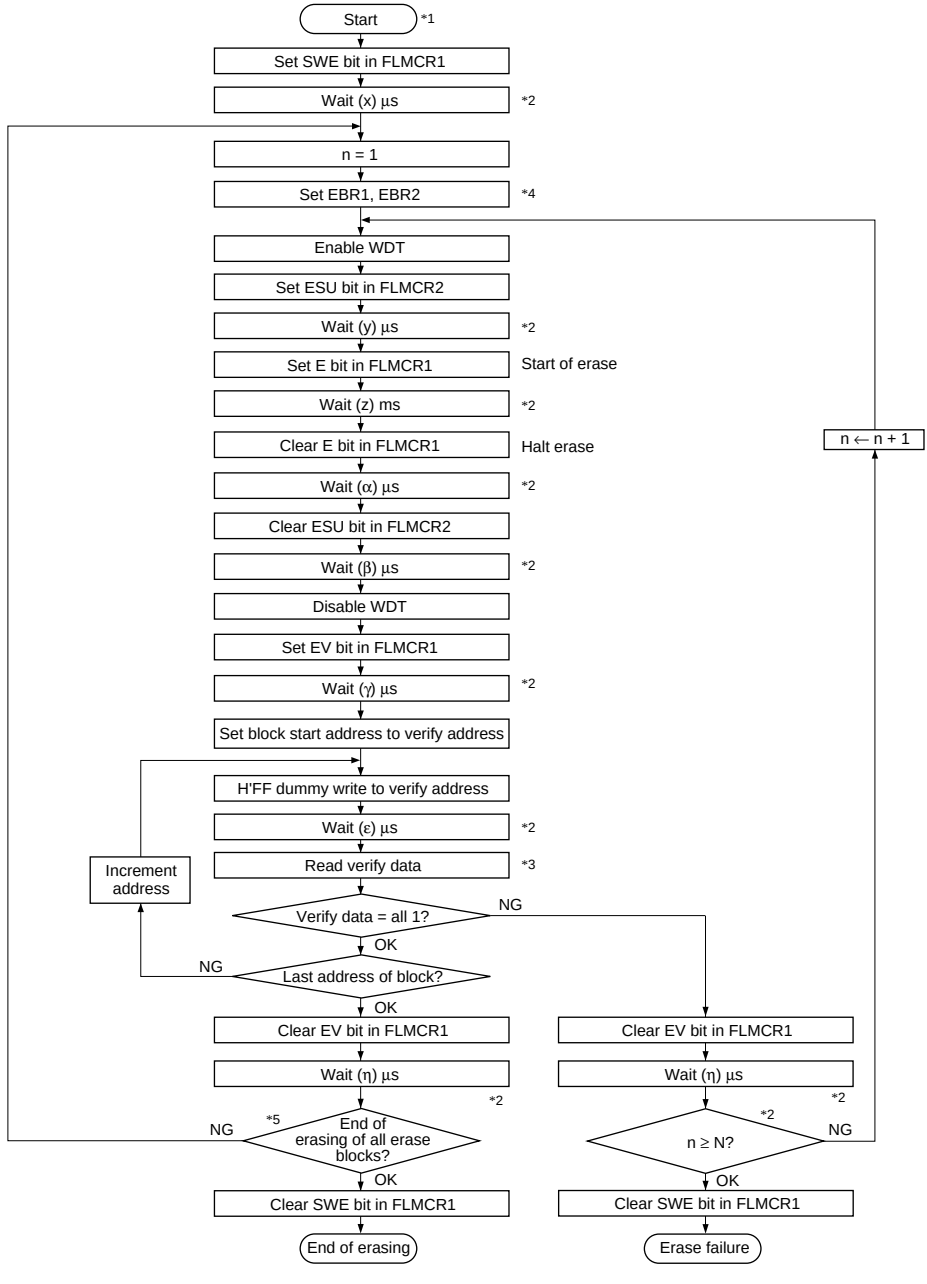
To perform data or program erasure, make a 1 bit setting for the flash memory area to be erased in EBR1 or EBR2 at least (x) µs after setting the SWE bit to 1 in FLMCR1. Next, the watchdog timer is set to prevent overerasing in the event of program runaway, etc. Set a value greater than (y + z + α + ß) µs as the WDT overflow period. After this, preparation for erase mode (erase setup) is carried out by setting the ESU bit in FLMCR2, and after the elapse of (y) µs or more, the operating mode is switched to erase mode by setting the E bit in FLMCR1. The time during which the E bit is set is the flash memory erase time. Ensure that the erase time does not exceed (z) ms.

Note: With flash memory erasing, preprogramming (setting all data in the memory to be erased to 0) is not necessary before starting the erase procedure.

### 17.9.4 Erase-Verify Mode

In erase-verify mode, data is read after memory has been erased to check whether it has been correctly erased.

After the elapse of the erase time, erase mode is exited (the E bit in FLMCR1 is cleared to 0, then the ESU bit in FLMCR2 is cleared to 0 at least (α) µs later), the watchdog timer is cleared after the elapse of (y + z + α + β) µs or more, and the operating mode is switched to erase-verify mode by setting the EV bit in FLMCR1. Before reading in erase-verify mode, a dummy write of H'FF data should be made to the addresses to be read. The dummy write should be executed after the elapse of (γ) µs or more. When the flash memory is read in this state (verify data is read in 16-bit units), the data at the latched address is read. Wait at least (ε) µs after the dummy write before performing this read operation. If the read data has been erased (all 1), a dummy write is performed to the next address, and erase-verify is performed. If the read data has not been erased, set erase mode again, and repeat the erase/erase-verify sequence in the same way. However, ensure that the erase/erase-verify sequence is not repeated more than (N) times. When verification is completed, exit erase-verify mode, and wait for at least (η) µs. If erasure has been completed on all the erase blocks, clear the SWE bit in FLMCR1 to 0. If there are any unerased blocks, make a 1 bit setting in EBR1 or EBR2 for the flash memory area to be erased, and repeat the erase/erase-verify sequence in the same way.




Notes: 1. Preprogramming (setting erase block data to all 0) is not necessary.
2. The values of x, y, z, α, β, γ, ε, η, and N are shown in section 20.1.6, Flash Memory Characteristics.
3. Verify data is read in 16-bit (word) units.
4. Set only one bit in EBR1or EBR2. More than one bit cannot be set.
5. Erasing is performed in block units. To erase a number of blocks, the individual blocks must be erased sequentially.

**Figure 17.22 Erase/Erase-Verify Flowchart (Single-Block Erase)**

## 17.10 Flash Memory Protection

There are three kinds of flash memory program/erase protection: hardware protection, software protection, and error protection.

### 17.10.1 Hardware Protection

Hardware protection refers to a state in which programming/erasing of flash memory is forcibly disabled or aborted. Hardware protection is reset by settings in flash memory control registers 1 and 2 (FLMCR1, FLMCR2) and erase block registers 1 and 2 (EBR1, EBR2). (See table 17.16.)

**Table 17.16 Hardware Protection**

| | | Functions | | |
|---|---|---|---|---|
| **Item** | **Description** | **Program** | **Erase** | **Verify*** |
| FWE pin protection | • When a low level is input to the FWE pin, FLMCR1, FLMCR2 (excluding the FLER bit), EBR1, and EBR2 are initialized, and the program/erase-protected state is entered. | No | No | No |
| Reset/standby protection | • In a reset (including a WDT overflow reset) and in standby mode, FLMCR1, FLMCR2, EBR1, and EBR2 are initialized, and the program/erase-protected state is entered.<br>• In a reset via the $\overline{\text{RES}}$ pin, the reset state is not entered unless the $\overline{\text{RES}}$ pin is held low until oscillation stabilizes after powering on. In the case of a reset during operation, hold the $\overline{\text{RES}}$ pin low for the $\overline{\text{RES}}$ pulse width ($t_{RESW}$) specified in the AC Characteristics section. | No | No | No |

Note: * Program verify and erase verify modes.

### 17.10.2 Software Protection

Software protection can be implemented by setting the SWE bit in FLMCR1, erase block registers 1 and 2 (EBR1, EBR2), and the RAMS bit in RAMER. When software protection is in effect, setting the P or E bit in flash memory control register 1 (FLMCR1) does not cause a transition to program mode or erase mode. (See table 17.17.)

**Table 17.17 Software Protection**

| Item | Description | Functions | | |
|---|---|---|---|---|
| | | Program | Erase | Verify* |
| SWE bit protection | • Clearing the SWE bit to 0 in FLMCR1 sets the program/erase-protected state for all blocks.<br>(Execute in on-chip RAM or external memory.) | No | No | No |
| Block specification protection | • Erase protection can be set for individual blocks by settings in erase block registers 1 and 2 (EBR1, EBR2).<br>However, write protection is disabled.<br>• Setting EBR1 and EBR2 to H'00 places all blocks in the erase-protected state. | — | No | Yes |
| Emulation protection | • Setting the RAMS bit to 1 in the RAM emulation register (RAMER) places all blocks in the program/erase-protected state. | No | No | Yes |

Note: * Program verify and erase verify modes.

### 17.10.3 Error Protection

In error protection, an error is detected when MCU runaway occurs during flash memory programming/erasing, or operation is not performed in accordance with the program/erase algorithm, and the program/erase operation is aborted. Aborting the program/erase operation prevents damage to the flash memory due to overprogramming or overerasing.

If the MCU malfunctions during flash memory programming/erasing, the FLER bit is set to 1 in FLMCR2 and the error protection state is entered. The FLMCR1, FLMCR2, EBR1, and EBR2 settings are retained, but program mode or erase mode is aborted at the point at which the error occurred. When the FLER bit is set to 1, it is not possible to re-enter the program mode or erase mode by resetting the P and E bits of FLMCR1. However, setting of the PV and EV bits of FLMCR1 is enabled, and a transition can be made to verify mode.

FLER bit setting conditions are as follows:

- When flash memory is read during programming/erasing (including a vector read or instruction fetch)
- Immediately after exception handling (excluding a reset) during programming/erasing
- When a SLEEP instruction (including software standby) is executed during programming/erasing
- When the CPU loses the bus during programming/erasing

Error protection is released only by a reset and in hardware standby mode.

Figure 17.23 shows the flash memory state transition diagram.

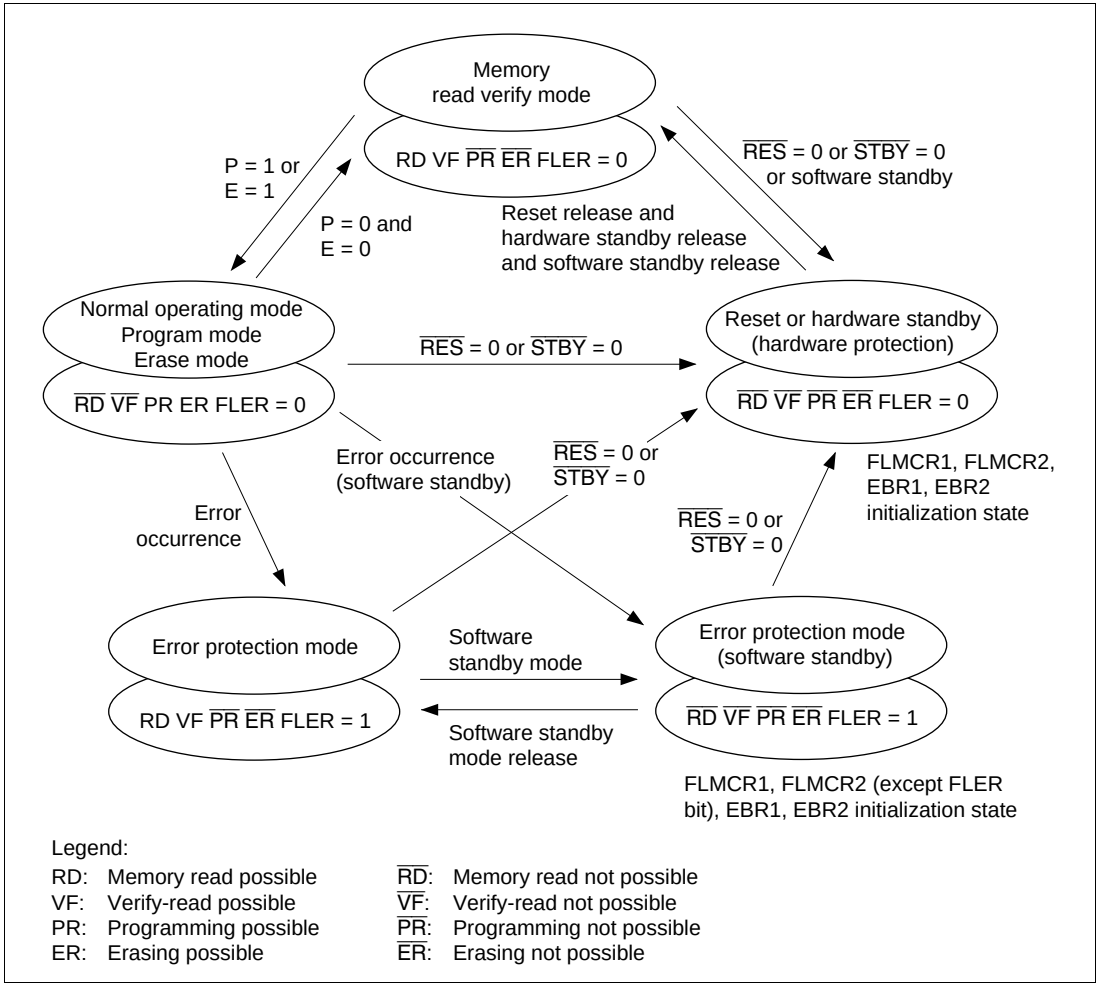



**Figure 17.23 Flash Memory State Transitions**

The error protect function has no effect on illegal operations unrelated to the setting conditions for the FLER bit. Also, if a significant amount of time has elapsed before the transition to the protect status, there is a possibility that the data in flash memory may already have become corrupted. Consequently, this function is not able to provide complete protection against corruption of the data in flash memory.

For this reason, it is necessary to run program and erase algorithms correctly while flash write enable (FWE) is being applied and to monitor the internal operation of the microcomputer for abnormalities using a watchdog timer, or the like, in order to prevent illegal operations of the sort mentioned above. Also, at the point when the transition is made to the protect mode, in some cases the flash memory may be in an erroneously programmed or erased status, or the programming or erasing may be incomplete due to a forced shutdown. In such a case, make sure to force a recovery (program rewrite) using the boot mode. Note that there may still be cases in which boot mode cannot be started normally due to excessive programming or erasing of the flash memory.

## 17.11 Flash Memory Emulation in RAM

### 17.11.1 Emulation in RAM

Since programming or erasing the flash memory takes time, it may be difficult to perform tuning by overwriting parameters and other data in real time. In such cases, making a setting in the RAM emulation register (RAMER) enables part of RAM to be overlapped onto the flash memory area so that data to be written to flash memory can be emulated in RAM in real time. After the RAMER setting has been made, accesses can be made from the flash memory area or the RAM area overlapping flash memory. Emulation can be performed in user mode and user program mode. Figure 17.24 shows an example of emulation of real-time flash memory programming.

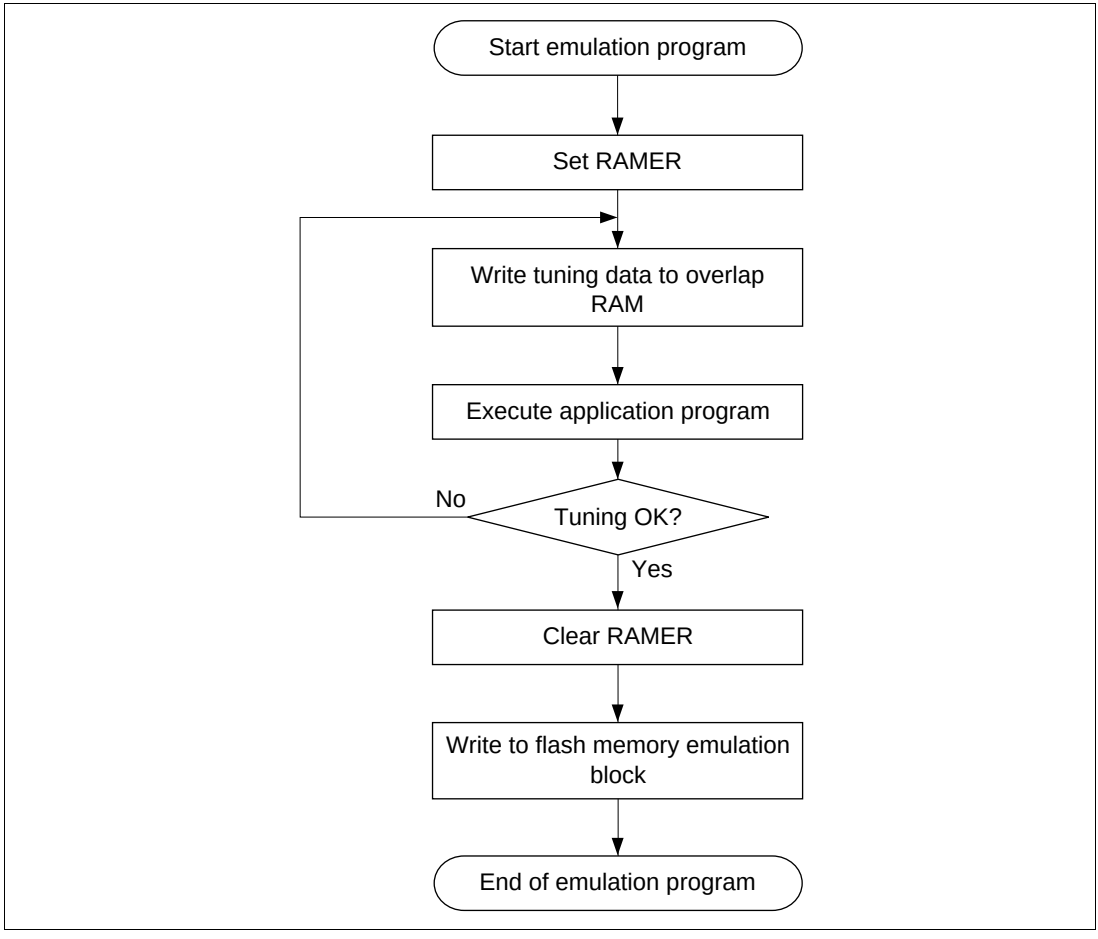



**Figure 17.24 Flowchart for Flash Memory Emulation in RAM**

### 17.11.2 RAM Overlap

An example in which flash memory block area EB1 is overlapped is shown below.

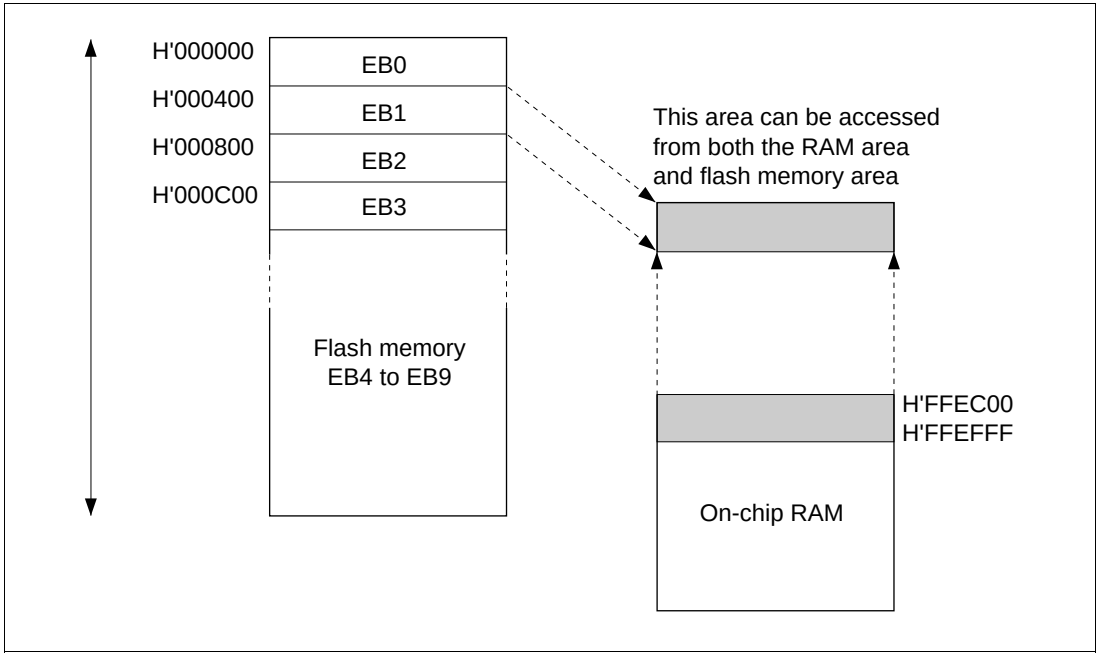



**Figure 17.25 Example of RAM Overlap Operation**

**Example in Which Flash Memory Block Area (EB1) is Overlapped**

1. Set bits RAMS, RAM1, and RAM0 in RAMER to 1, 0, 1, to overlap part of RAM onto the area (EB1) for which real-time programming is required.
2. Real-time programming is performed using the overlapping RAM.
3. After the program data has been confirmed, the RAMS bit is cleared, releasing RAM overlap.
4. The data written in the overlapping RAM is written into the flash memory space (EB1).

Notes: 1. When the RAMS bit is set to 1, program/erase protection is enabled for all blocks regardless of the value of RAM1 and RAM0 (emulation protection). In this state, setting the P or E bit in flash memory control register 1 (FLMCR1) will not cause a transition to program mode or erase mode. When actually programming a flash memory area, the RAMS bit should be cleared to 0.

2. A RAM area cannot be erased by execution of software in accordance with the erase algorithm while flash memory emulation in RAM is being used.
3. Block area EB0 includes the vector table. When performing RAM emulation, the vector table is needed by the overlap RAM.

## 17.12 Interrupt Handling when Programming/Erasing Flash Memory

All interrupts, including NMI interrupt is disabled when flash memory is being programmed or erased (when the P or E bit is set in FLMCR1), and while the boot program is executing in boot mode*[1], to give priority to the program or erase operation. There are three reasons for this:

1. Interrupt during programming or erasing might cause a violation of the programming or erasing algorithm, with the result that normal operation could not be assured.
2. In the interrupt exception handling sequence during programming or erasing, the vector would not be read correctly*[2], possibly resulting in MCU runaway.
3. If interrupt occurred during boot program execution, it would not be possible to execute the normal boot mode sequence.

For these reasons, in on-board programming mode alone there are conditions for disabling interrupt, as an exception to the general rule. However, this provision does not guarantee normal erasing and programming or MCU operation. All requests, including NMI interrupt, must therefore be restricted inside and outside the MCU when programming or erasing flash memory. NMI interrupt is also disabled in the error-protection state while the P or E bit remains set in FLMCR1.

Notes: 1. Interrupt requests must be disabled inside and outside the MCU until the programming control program has completed programming.

2. The vector may not be read correctly in this case for the following two reasons:
   - If flash memory is read while being programmed or erased (while the P or E bit is set in FLMCR1), correct read data will not be obtained (undetermined values will be returned).
   - If the interrupt entry in the vector table has not been programmed yet, interrupt exception handling will not be executed correctly.

## 17.13 Flash Memory Writer Mode

### 17.13.1 Writer Mode Setting

Programs and data can be written and erased in writer mode as well as in the on-board programming modes. In writer mode, the on-chip ROM can be freely programmed using a PROM programmer that supports Hitachi microcomputer device types with 128-kbyte on-chip flash memory. In writer mode, the on-chip ROM can be freely programmed using a PROM programmer that supports the Hitachi microcomputer device type with 128-kbyte on-chip flash memory. Flash memory read mode, auto-program mode, auto-erase mode, and status read mode are supported with this device type. In auto-program mode, auto-erase mode, and status read mode, a status polling procedure is used, and in status read mode, detailed internal signals are output after execution of an auto-program or auto-erase operation.

Table 17.18 shows writer mode pin settings.

**Table 17.18 Writer Mode Pin Settings**

| Pin Names | Settings/External Circuit Connection |
|---|---|
| Mode pins: $MD_2$, $MD_1$, $MD_0$ | Low-level input |
| Mode setting pins: $PF_2$, $PF_1$, $PF_0$ | High-level input to $PF_2$, low-level input to $PF_1$ and $PF_0$ |
| FWE pin | High-level input (in auto-program and auto-erase modes) |
| $\overline{\text{STBY}}$ pin | High-level input (do not select hardware standby mode) |
| $\overline{\text{RES}}$ pin | Power-on reset circuit |
| NMI pin | High-level input (for power-on reset) |
| XTAL, EXTAL pins | Oscillator circuit |
| Other pins requiring setting: $P2_3$, $P2_5$ | High-level input to $P2_3$ and $P2_5$ |

### 17.13.2 Socket Adapters and Memory Map

In writer mode, a socket adapter is mounted on the writer programmer to match the package concerned. Socket adapters are available for each writer manufacturer supporting the Hitachi microcomputer device type with 128-kbyte on-chip flash memory. The model numbers of compatible socket adapters are listed in table 17.19.

Figure 17.26 shows socket adapter pin correspondences and figure 17.26 shows the memory map in writer mode. For pin names in writer mode, see section 1.3.2, Pin Functions in Each Operating Mode.

**Table 17.19 Socket Adapter Name**

| | | Socket Adapter Name | |
|---|---|---|---|
| **Product Model** | **Package Name** | **Minato Electronics** | **Data I/O Japan** |
| HD64F2345 | 100-pin TQFP (TFP-100B) | ME2345ESNF1H | HF234BT100D3201 |
| | 100-pin TQFP (TFP-100G) | ME2345ESMF1H | HF234GT100D3201 |
| | 100-pin QFP (FP-100A) | ME2345ESFF1H | HF234AQ100D3201 |
| | 100-pin QFP (FP-100B) | ME2345ESHF1H | HF234BQ100D3201 |

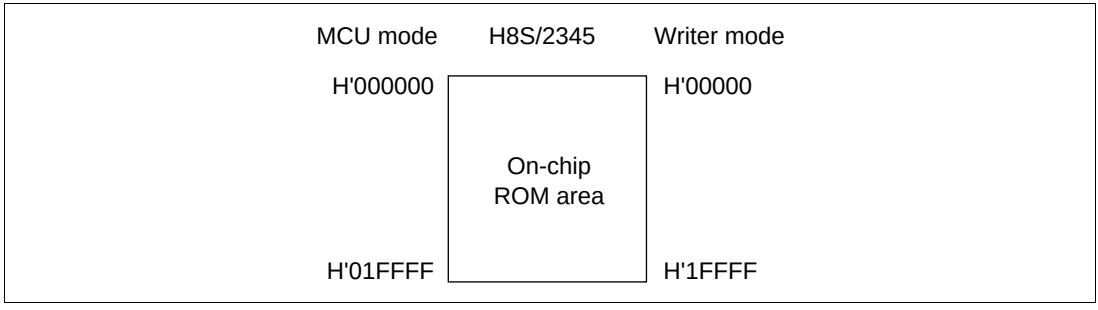



**Figure 17.26 Memory Map in Writer Mode**

### 17.13.3 Writer Mode Operation

Table 17.20 shows how the different operating modes are set when using writer mode, and table 17.21 lists the commands used in writer mode. Details of each mode are given below.

- Memory Read Mode

  Memory read mode supports byte reads.
- Auto-Program Mode

  Auto-program mode supports programming of 128 bytes at a time. Status polling is used to confirm the end of auto-programming.
- Auto-Erase Mode

  Auto-erase mode supports automatic erasing of the entire flash memory. Status polling is used to confirm the end of auto-erasing.
- Status Read Mode

  Status polling is used for auto-programming and auto-erasing, and normal termination can be confirmed by reading the $FO_6$ signal. In status read mode, error information is output if an error occurs.

**Table 17.20 Settings for Each Operating Mode in Writer Mode**

| | Pin Names | | | | | |
|---|---|---|---|---|---|---|
| **Mode** | **FWE** | **$\overline{CE}$** | **$\overline{OE}$** | **$\overline{WE}$** | **$FO_0$ to $FO_7$** | **$FA_0$ to $FA_{16}$** |
| Read | H or L | L | L | H | Data output | Ain |
| Output disable | H or L | L | H | H | Hi-z | X |
| Command write | H or L*[3] | L | H | L | Data input | Ain*[2] |
| Chip disable*[1] | H or L | H | X | X | Hi-z | X |

Legend:

H: High level

L: Low level

X: Don't care

Hi-z: High impedance

Notes: 1. Chip disable is not a standby state; internally, it is an operation state.

2. Ain indicates that there is also address input in auto-program mode.

3. For command writes when making a transition to auto-program or auto-erase mode, input a high level to the FWE pin.

**Table 17.21 Writer Mode Commands**

| | Number | 1st Cycle | | | 2nd Cycle | | |
|---|---|---|---|---|---|---|---|
| **Command Name** | **of Cycles** | **Mode** | **Address** | **Data** | **Mode** | **Address** | **Data** |
| Memory read mode | 1 + n*[1] | Write | X | H'00 | Read | RA | Dout |
| Auto-program mode | 129*[2] | Write | X | H'40 | Write | WA | Din |
| Auto-erase mode | 2 | Write | X | H'20 | Write | X | H'20 |
| Status read mode | 2 | Write | X | H'71 | Write | X | H'71 |

Legend:

RA: Read address

WA: Program address (Write address)

Dout: Read data

Din: Program data

Notes: 1. In memory read mode, the number of cycles depends on the number of address write cycles (n).

2. In auto-program mode. 129 cycles are required for command writing by a simultaneous 128-byte write.

**Table 17.22 DC Characteristics in Memory Read Mode**
**(Conditions: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $T_a$ = 25°C ±5°C)**

| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|
| Input high-level voltage | $FO_7$–$FO_0$, $FA_{16}$–$FA_0$ | $V_{IH}$ | 2.2 | — | $V_{CC}$ + 0.3 | V | |
| Input low-level voltage | $FO_7$–$FO_0$, $FA_{16}$–$FA_0$ | $V_{IL}$ | 0.3 | — | 0.8 | V | |
| Schmitt trigger input voltage | $\overline{OE}$, $\overline{CE}$, $\overline{WE}$ | $V_{T-}$ | 1.0 | — | 2.5 | V | |
| | | $V_{T+}$ | 2.0 | — | 3.5 | V | |
| | | $V_{T+}$–$V_{T-}$ | 0.4 | — | — | V | |
| Output high-level voltage | $FO_7$–$FO_0$ | $V_{OH}$ | 2.4 | — | — | V | $I_{OH}$ = –200 µA |
| Output low-level voltage | $FO_7$–$FO_0$ | $V_{OL}$ | — | — | 0.45 | V | $I_{OL}$ = 1.6 mA |
| Input leak current | $FO_7$–$FO_0$, $FA_{16}$–$FA_0$ | $\lvert I_{LI} \rvert$ | — | — | 2 | µA | |
| $V_{CC}$ current | During read | $I_{CC}$ | — | 40 | 65 | mA | |
| | During programming | $I_{CC}$ | — | 50 | 85 | mA | |
| | During erasing | $I_{CC}$ | — | 50 | 85 | mA | |

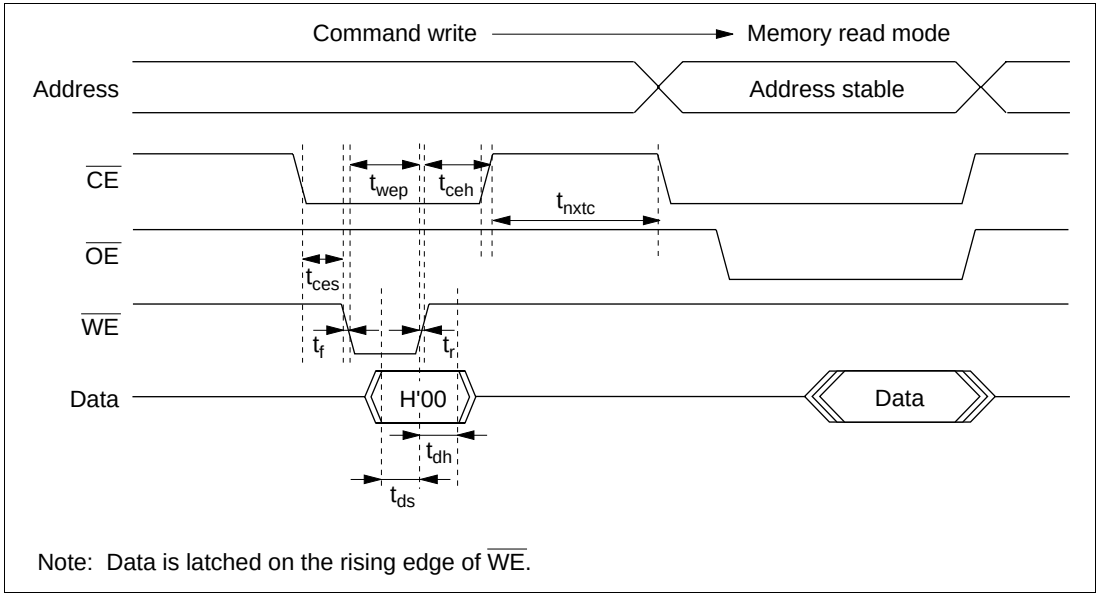
Note: Refer to the maximum rating for the F-ZTAT version “20.1.1 Absolute Maximum Ratings.” If the maximum rating is exceeded, the LSI may be damaged permanently.

### 17.13.4 Memory Read Mode

- After the end of an auto-program, auto-erase, or status read operation, the command wait state is entered. To read memory contents, a transition must be made to memory read mode by means of a command write before the read is executed.
- Command writes can be performed in memory read mode, just as in the command wait state.
- Once memory read mode has been entered, consecutive reads can be performed.
- After power-on, memory read mode is entered.

**Table 17.23 AC Characteristics in Memory Read Mode Transition AC Characteristics (Conditions: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $T_a$ = 25°C ±5°C)**

| Item | Symbol | Min | Max | Unit | Notes |
|---|---|---|---|---|---|
| Command write cycle | $t_{nxtc}$ | 20 | | µs | |
| $\overline{CE}$ hold time | $t_{ceh}$ | 0 | | ns | |
| $\overline{CE}$ setup time | $t_{ces}$ | 0 | | ns | |
| Data hold time | $t_{dh}$ | 50 | | ns | |
| Data setup time | $t_{ds}$ | 50 | | ns | |
| Write pulse width | $t_{wep}$ | 70 | | ns | |
| $\overline{WE}$ rise time | $t_r$ | | 30 | ns | |
| $\overline{WE}$ fall time | $t_f$ | | 30 | ns | |




Note: Data is latched on the rising edge of $\overline{WE}$.

**Figure 17.27 Memory Read Mode Transition Timing Waveforms**

**Table 17.24 AC Characteristics in Memory Read Mode**
**(Conditions: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $T_a$ = 25°C ±5°C)**

| Item | Symbol | Min | Max | Unit | Notes |
|---|---|---|---|---|---|
| Access time | $t_{acc}$ | | 20 | µs | |
| $\overline{CE}$ output delay time | $t_{ce}$ | | 150 | ns | |
| $\overline{OE}$ output delay time | $t_{oe}$ | | 150 | ns | |
| Output disable delay time | $t_{df}$ | | 100 | ns | |
| Data output hold time | $t_{oh}$ | 5 | | ns | |

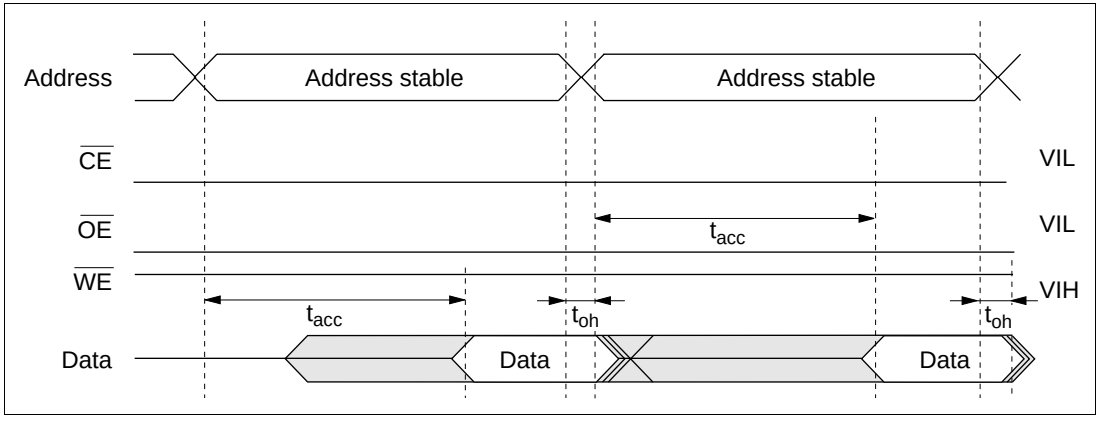



**Figure 17.28 Timing Waveforms for $\overline{CE}$/$\overline{OE}$ Enable State Read**

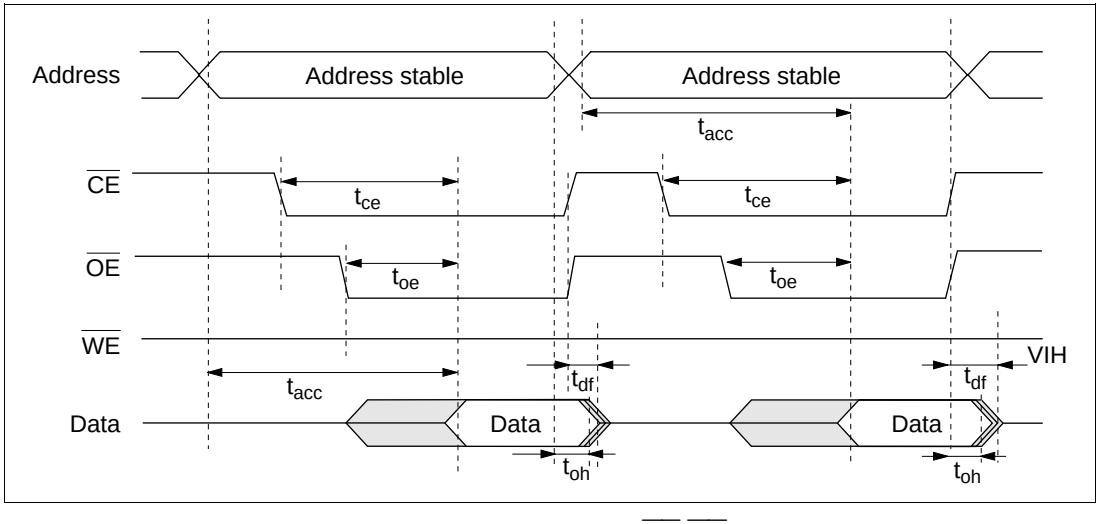



**Figure 17.29 Timing Waveforms for $\overline{CE}$/$\overline{OE}$ Clocked Read**

**Table 17.25 AC Characteristics when Entering Another Mode from Memory Read Mode (Conditions: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $T_a$ = 25°C ±5°C)**

| Item | Symbol | Min | Max | Unit | Notes |
|---|---|---|---|---|---|
| Command write cycle | $t_{nxtc}$ | 20 | | µs | |
| $\overline{CE}$ hold time | $t_{ceh}$ | 0 | | ns | |
| $\overline{CE}$ setup time | $t_{ces}$ | 0 | | ns | |
| Data hold time | $t_{dh}$ | 50 | | ns | |
| Data setup time | $t_{ds}$ | 50 | | ns | |
| Write pulse width | $t_{wep}$ | 70 | | ns | |
| $\overline{WE}$ rise time | $t_r$ | | 30 | ns | |
| $\overline{WE}$ fall time | $t_f$ | | 30 | ns | |

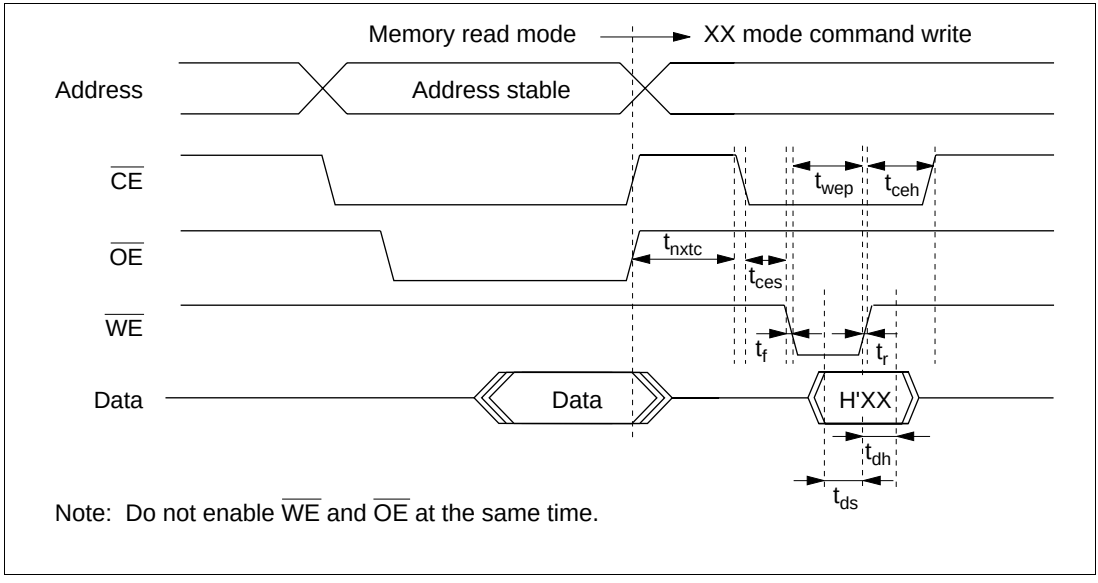



**Figure 17.30 Timing Waveforms when Entering Another Mode from Memory Read Mode**

### 17.13.5 Auto-Program Mode

**AC Characteristics:** The auto-program mode supports the writing of 128 bytes simultaneously.

**Table 17.26 AC Characteristics in Auto-Program Mode**
**(Conditions: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $T_a$ = 25°C ±5°C)**

| Item | Symbol | Min | Max | Unit | Notes |
|---|---|---|---|---|---|
| Command write cycle | $t_{nxtc}$ | 20 | | µs | |
| $\overline{CE}$ hold time | $t_{ceh}$ | 0 | | ns | |
| $\overline{CE}$ setup time | $t_{ces}$ | 0 | | ns | |
| Data hold time | $t_{dh}$ | 50 | | ns | |
| Data setup time | $t_{ds}$ | 50 | | ns | |
| Write pulse width | $t_{wep}$ | 70 | | ns | |
| Status polling start time | $t_{wsts}$ | 1 | | ms | |
| Status polling access time | $t_{spa}$ | | 150 | ns | |
| Address setup time | $t_{as}$ | 0 | | ns | |
| Address hold time | $t_{ah}$ | 60 | | ns | |
| Memory write time | $t_{write}$ | 1 | 3000 | ms | |
| $\overline{WE}$ rise time | $t_r$ | | 30 | ns | |
| $\overline{WE}$ fall time | $t_f$ | | 30 | ns | |
| Write setup time | $t_{pns}$ | 100 | | ns | |
| Write end setup time | $t_{pnh}$ | 100 | | ns | |

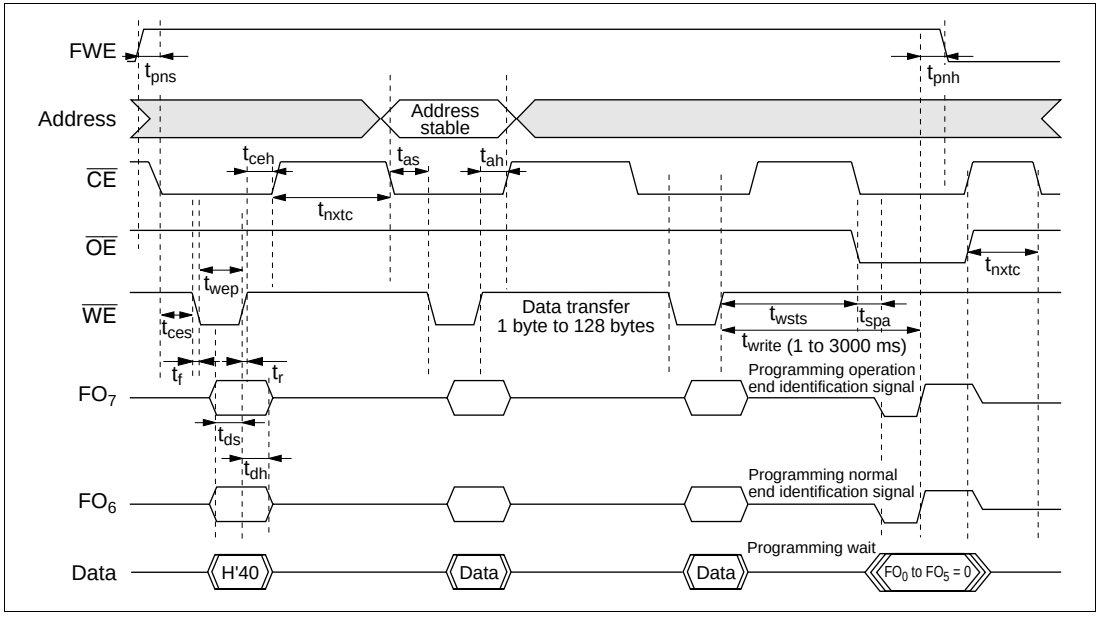



**Figure 17.31 Auto-Program Mode Timing Waveforms**

**Notes on Use of Auto-Program Mode**

- In auto-program mode, 128 bytes are programmed simultaneously. This should be carried out by executing 128 consecutive byte transfers.
- A 128-byte data transfer is necessary even when programming fewer than 128 bytes. In this case, H'FF data must be written to the extra addresses.
- The lower 8 bits of the transfer address must be H'00 or H'80. If a value other than an effective address is input, processing will switch to a memory write operation but a write error will be flagged.
- Memory address transfer is performed in the second cycle (figure 17.31). Do not perform memory address transfer after the second cycle.
- Do not perform a command write during a programming operation.
- Perform one auto-programming operation for a 128-byte block for each address. Characteristics are not guaranteed for two or more programming operations.
- Confirm normal end of auto-programming by checking $FO_6$. Alternatively, status read mode can also be used for this purpose ($FO_7$ status polling uses the auto-program operation end identification pin).
- The status polling $FO_6$ and $FO_7$ pin information is retained until the next command write. Until the next command write is performed, reading is possible by enabling $\overline{CE}$ and $\overline{OE}$.

### 17.13.6 Auto-Erase Mode

Autro-erase mode supports only automatic erasure of the entire flash memory mat.

**AC Characteristics**

**Table 17.27 AC Characteristics in Auto-Erase Mode (Conditions: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $T_a$ = 25°C ±5°C)**

| Item | Symbol | Min | Max | Unit | Notes |
|---|---|---|---|---|---|
| Command write cycle | $t_{nxtc}$ | 20 | | µs | |
| $\overline{CE}$ hold time | $t_{ceh}$ | 0 | | ns | |
| $\overline{CE}$ setup time | $t_{ces}$ | 0 | | ns | |
| Data hold time | $t_{dh}$ | 50 | | ns | |
| Data setup time | $t_{ds}$ | 50 | | ns | |
| Write pulse width | $t_{wep}$ | 70 | | ns | |
| Status polling start time | $t_{ests}$ | 1 | | ms | |
| Status polling access time | $t_{spa}$ | | 150 | ns | |
| Memory erase time | $t_{erase}$ | 100 | 40000 | ms | |
| $\overline{WE}$ rise time | $t_r$ | | 30 | ns | |
| $\overline{WE}$ fall time | $t_f$ | | 30 | ns | |
| Erase setup time | $t_{ens}$ | 100 | | ns | |
| Erase end setup time | $t_{enh}$ | 100 | | ns | |

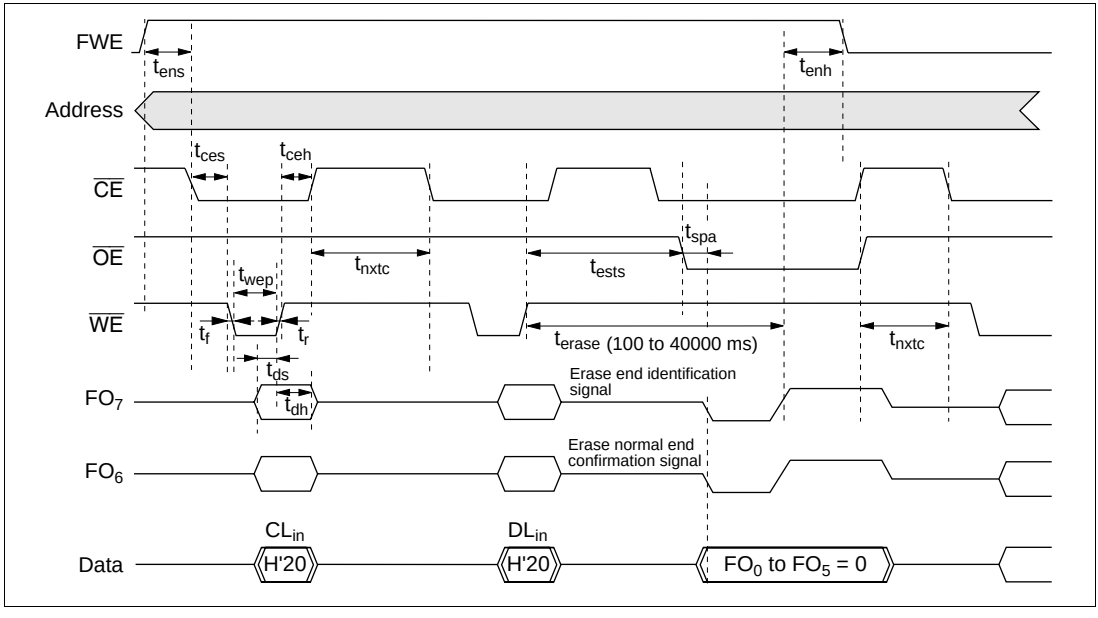



**Figure 17.32 Auto-Erase Mode Timing Waveforms**

**Notes on Use of Erase-Program Mode**

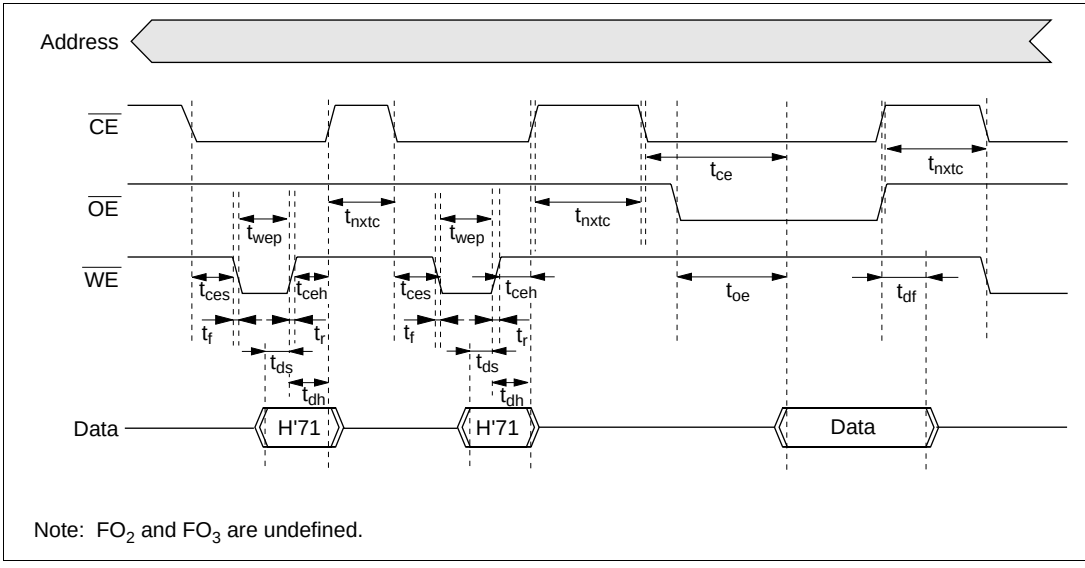
- Auto-erase mode supports only entire memory erasing.
- Do not perform a command write during auto-erasing.
- Confirm normal end of auto-erasing by checking $FO_6$. Alternatively, status read mode can also be used for this purpose ($FO_7$ status polling uses the auto-erase operation end identification pin).
- The status polling $FO_6$ and $FO_7$ pin information is retained until the next command write. Until the next command write is performed, reading is possible by enabling $\overline{CE}$ and $\overline{OE}$.

### 17.13.7 Status Read Mode

- Status read mode is used to identify what type of abnormal end has occurred. Use this mode when an abnormal end occurs in auto-program mode or auto-erase mode.
- The return code is retained until a command write for other than status read mode is performed.

**Table 17.28 AC Characteristics in Status Read Mode**
**(Conditions: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $T_a$ = 25°C ±5°C)**

| Item | Symbol | Min | Max | Unit | Notes |
|---|---|---|---|---|---|
| Command write cycle | $t_{nxtc}$ | 20 | | µs | |
| $\overline{CE}$ hold time | $t_{ceh}$ | 0 | | ns | |
| $\overline{CE}$ setup time | $t_{ces}$ | 0 | | ns | |
| Data hold time | $t_{dh}$ | 50 | | ns | |
| Data setup time | $t_{ds}$ | 50 | | ns | |
| Write pulse width | $t_{wep}$ | 70 | | ns | |
| $\overline{OE}$ output delay time | $t_{oe}$ | | 150 | ns | |
| Disable delay time | $t_{df}$ | | 100 | ns | |
| $\overline{CE}$ output delay time | $t_{ce}$ | | 150 | ns | |
| $\overline{WE}$ rise time | $t_r$ | | 30 | ns | |
| $\overline{WE}$ fall time | $t_f$ | | 30 | ns | |




Note: $FO_2$ and $FO_3$ are undefined.

**Figure 17.33 Status Read Mode Timing Waveforms**

**Table 17.29 Status Read Mode Return Commands**

| Pin Name | $FO_7$ | $FO_6$ | $FO_5$ | $FO_4$ | $FO_3$* | $FO_2$* | $FO_1$ | $FO_0$ |
|---|---|---|---|---|---|---|---|---|
| Attribute | Normal end identification | Command error | Program-ming error | Erase error | — | — | Program-ming or erase count exceeded | Effective address error |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Indications | Normal end: 0<br>Abnormal end: 1 | Command error: 1<br>Otherwise: 0 | Program-ming error: 1<br>Otherwise: 0 | Erase error: 1<br>Otherwise: 0 | — | — | Count exceeded: 1<br>Otherwise: 0 | Effective address error: 1<br>Otherwise: 0 |

Note: * $FO_2$ and $FO_3$ are undefined.

**Status Read Mode Usage Note**: After the auto-program mode or auto-erase mode has completed, make sure to enter the status read mode before powering off the system.

The return commands are undefined immediately after power-on or if the system has been powered off once.

### 17.13.8 Status Polling

- The $FO_7$ status polling flag indicates the operating status in auto-program or auto-erase mode.
- The $FO_6$ status polling flag indicates a normal or abnormal end in auto-program or auto-erase mode.

**Table 17.30 Status Polling Output Truth Table**

| Pin Names | Internal Operation in Progress | Abnormal End | — | Normal End |
|---|---|---|---|---|
| $FO_7$ | 0 | 1 | 0 | 1 |
| $FO_6$ | 0 | 0 | 1 | 1 |
| $FO_0$ to $FO_5$ | 0 | 0 | 0 | 0 |

### 17.13.9 Writer Mode Transition Time

Commands cannot be accepted during the oscillation stabilization period or the writer mode setup period. After the writer mode setup time, a transition is made to memory read mode.

**Table 17.31 Command Wait State Transition Time Specifications**

| Item | Symbol | Min | Max | Unit | Notes |
|---|---|---|---|---|---|
| Standby release (oscillation stabilization time) | $t_{osc1}$ | 10 | — | ms | |
| Writer mode setup time | $t_{bmv}$ | 10 | — | ms | |
| $V_{CC}$ hold time | $t_{dwn}$ | 0 | — | ms | |

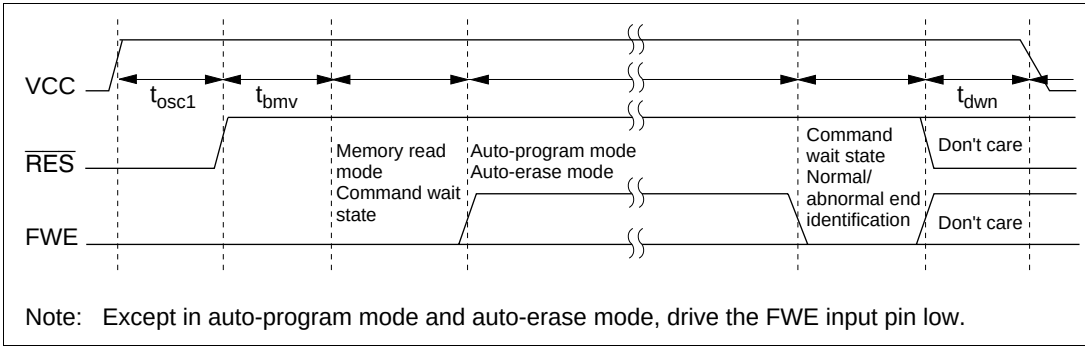



**Figure 17.34 Oscillation Stabilization Time, Writer Mode Setup Time, and Power Supply Fall Sequence**

### 17.13.10 Notes On Memory Programming

- When programming addresses which have previously been programmed, carry out auto-erasing before auto-programming. (See figure 17.35.)
- When performing programming using writer mode on a chip that has been programmed/erased in an on-board programming mode, auto-erasing is recommended before carrying out auto-programming.

Notes: 1. The flash memory is initially in the erased state when the device is shipped by Hitachi. For other chips for which the erasure history is unknown, it is recommended that auto-erasing be executed to check and supplement the initialization (erase) level.

2. Auto-programming in the writer mode should be performed only once for each 128-byte write unit block.

   It is not possible to write additional data to a 128-byte write unit block that has already been programmed. To reprogram a block, first use the auto-erase mode and then use the auto-program mode.

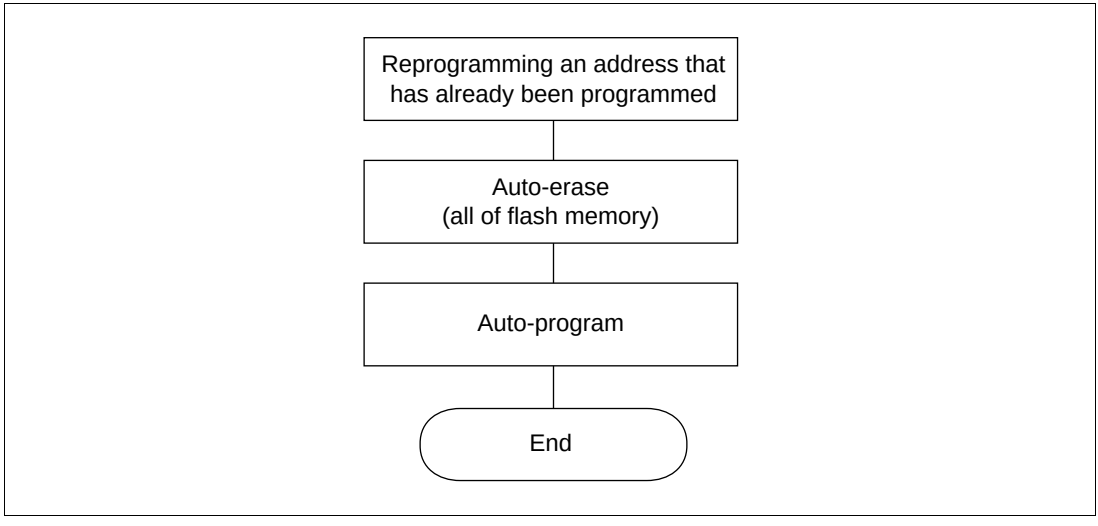



**Figure 17.35 Reprogramming an Address that has Already Been Programmed**

## 17.14 Flash Memory Programming and Erasing Precautions

Precautions concerning the use of on-board programming mode, the RAM emulation function, and writer mode are summarized below.

**Use the specified voltages and timing for programming and erasing:** Applied voltages in excess of the rating can permanently damage the device. Use a PROM programmer that supports Hitachi microcomputer device types with 128-kbyte on-chip flash memory.

Do not select the HN28F101 setting for the PROM programmer, and only use the specified socket adapter. Incorrect use will result in damaging the device.

**Powering on and off (see figures 17.36 to 17.38):** Do not apply a high level to the FWE pin until $V_{CC}$ has stabilized. Also, drive the FWE pin low before turning off $V_{CC}$.

When applying or disconnecting $V_{CC}$, fix the FWE pin low and place the flash memory in the hardware protection state.

The power-on and power-off timing requirements should also be satisfied in the event of a power failure and subsequent recovery. If these timing requirements are not satisfied, the microcomputer experience program runaway, possibly resulting in excessive programming and erasing that could cause the memory cell to no longer operate properly.

**FWE pin application/disconnection (see figure 19.36 to figure 19.38):** FWE pin application should be carried out when MCU operation is in a stable condition. If MCU operation is not stable, fix the FWE pin low and set the protection state.

The following points must be observed concerning FWE pin application and disconnection to prevent unintentional programming or erasing of flash memory:

- Apply the FWE pin when the $V_{CC}$ voltage has stabilized within its rated voltage range. Apply the FWE pin when oscillation has stabilized (after the oscillation settling time $t_{OSC1}$ has elapsed).
- In boot mode, apply and disconnect the FWE pin during a reset.
- In user program mode, the FWE pin can be switched between high and low level regardless of the reset state. FWE pin input can also be switched during program execution in flash memory.
- Do not apply the FWE pin if program runaway has occurred.
- Disconnect the FWE pin only when the SWE, ESU, PSU, EV, PV, P, and E bits in FLMCR1 and FLMCR2 are cleared.
- Make sure that the SWE, ESU, PSU, EV, PV, P, and E bits are not set by mistake when applying or disconnecting the FWE pin.

**Do not apply a constant high level to the FWE pin:** The only time a high level should be applied to the FWE pin in order to prevent erroneous programming or erasing due to program runaway is when programming or erasing flash memory (including when RAM is being used to emulate flash memory). A system configuration in which a high level is constantly applied to the FWE should be avoided. Also, while a high level is applied to the FWE pin, the watchdog timer should be activated to prevent overprogramming or overerasing due to program runaway, etc.

**Use the recommended algorithm when programming and erasing flash memory:** The recommended algorithm enables programming and erasing to be carried out without subjecting the device to voltage stress or sacrificing program data reliability. When setting the P or E bit in FLMCR1, the watchdog timer should be set beforehand as a precaution against program runaway, etc.

**Do not set or clear the SWE bit during program execution in flash memory:** Clear the SWE bit before executing a program or reading data in flash memory. When the SWE bit is set, data in flash memory can be rewritten, but flash memory should only be accessed for verify operations (verification during programming/erasing). Similarly, when using the RAM emulation function while a high level is being input to the FWE pin, the SWE bit must be cleared before executing a program or reading data in flash memory. However, the RAM area overlapping flash memory space can be read and written to regardless of whether the SWE bit is set or cleared.

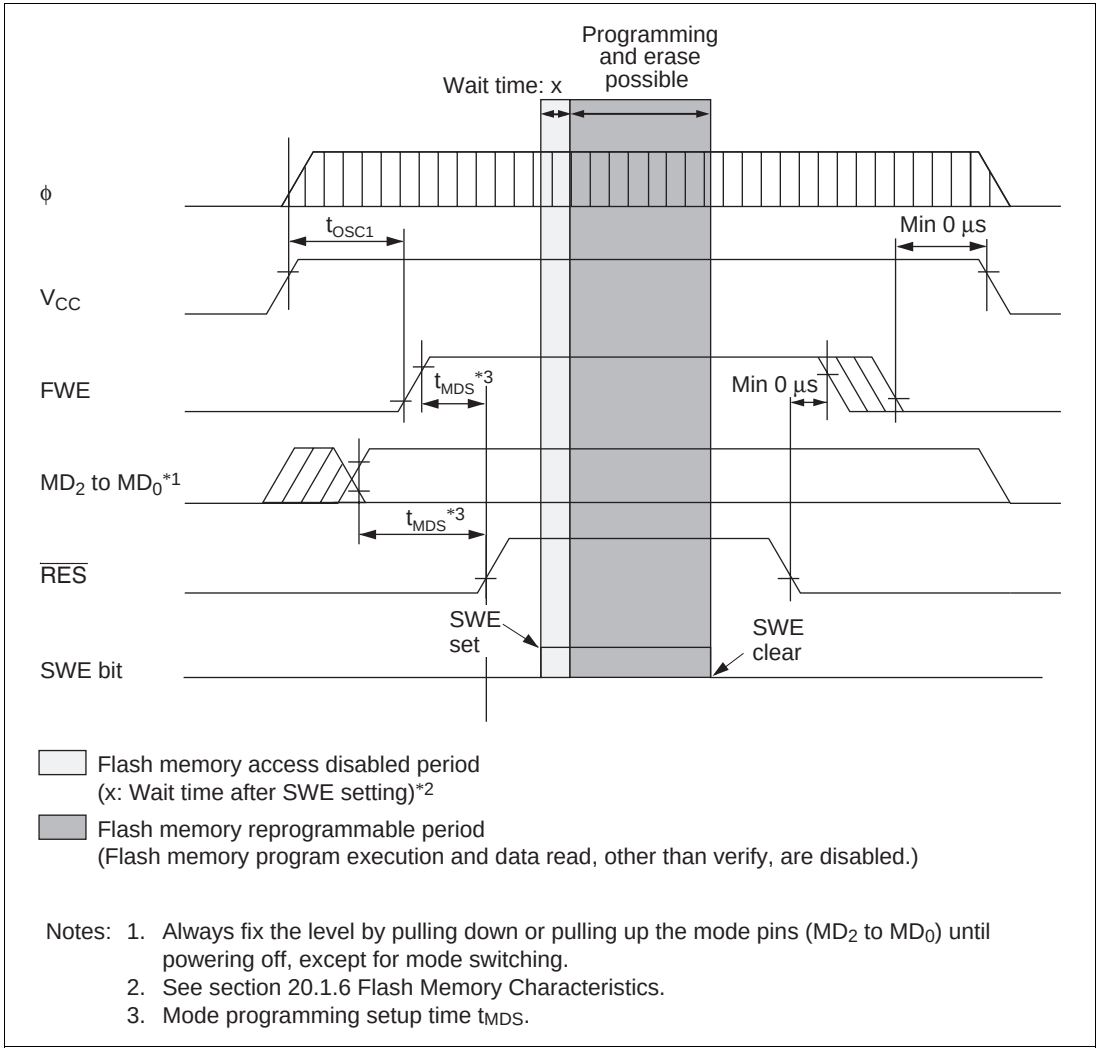
**Do not use interrupts while flash memory is being programmed or erased:** All interrupt requests, including NMI, should be disabled during FWE application to give priority to program/erase operations (including when RAM is being used to emulate flash memory).

Also, it is necessary to prohibit release of the bus.

**Do not perform additional programming. Erase the memory before reprogramming.** In on-board programming, perform only one programming operation on a 32-byte programming unit block. In writer mode, too, perform only one programming operation on a 128-byte programming unit block. Programming should be carried out with the entire programming unit block erased.

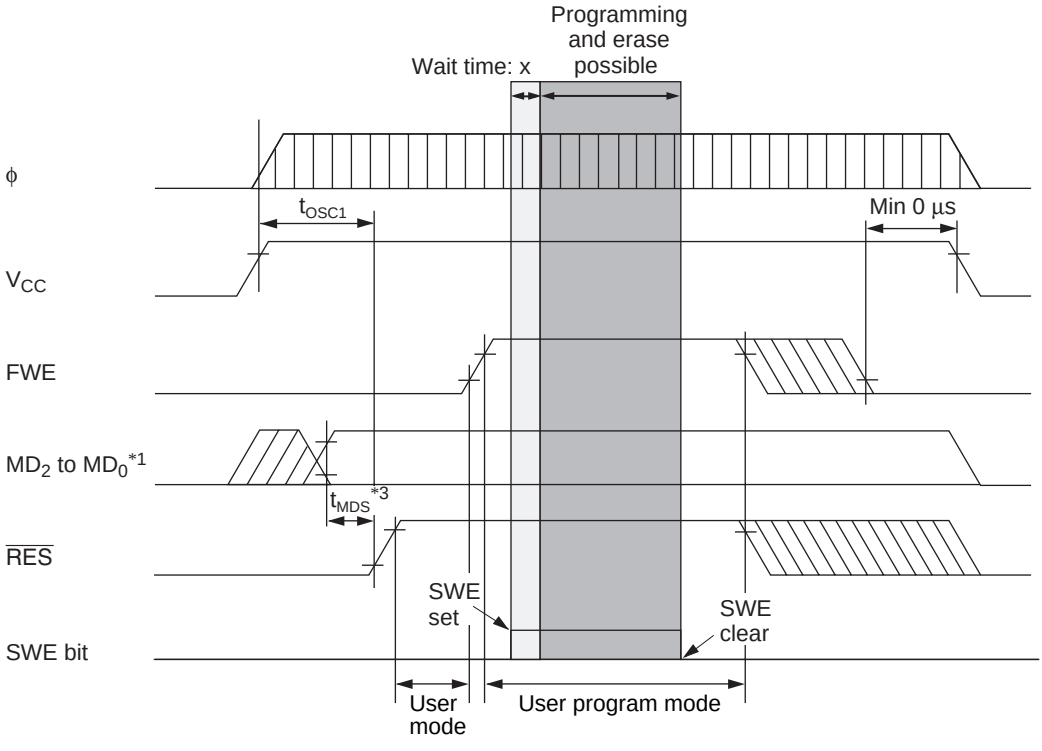
**Before programming, check that the chip is correctly mounted in the PROM programmer.** Overcurrent damage to the device can result if the index marks on the PROM programmer socket, socket adapter, and chip are not correctly aligned.

**Do not touch the socket adapter or chip during programming.** Touching either of these can cause contact faults and write errors.




Notes: 1. Always fix the level by pulling down or pulling up the mode pins ($MD_2$ to $MD_0$) until powering off, except for mode switching.
2. See section 20.1.6 Flash Memory Characteristics.
3. Mode programming setup time $t_{MDS}$.

**Figure 17.36 Powering On/Off Timing (Boot Mode)**

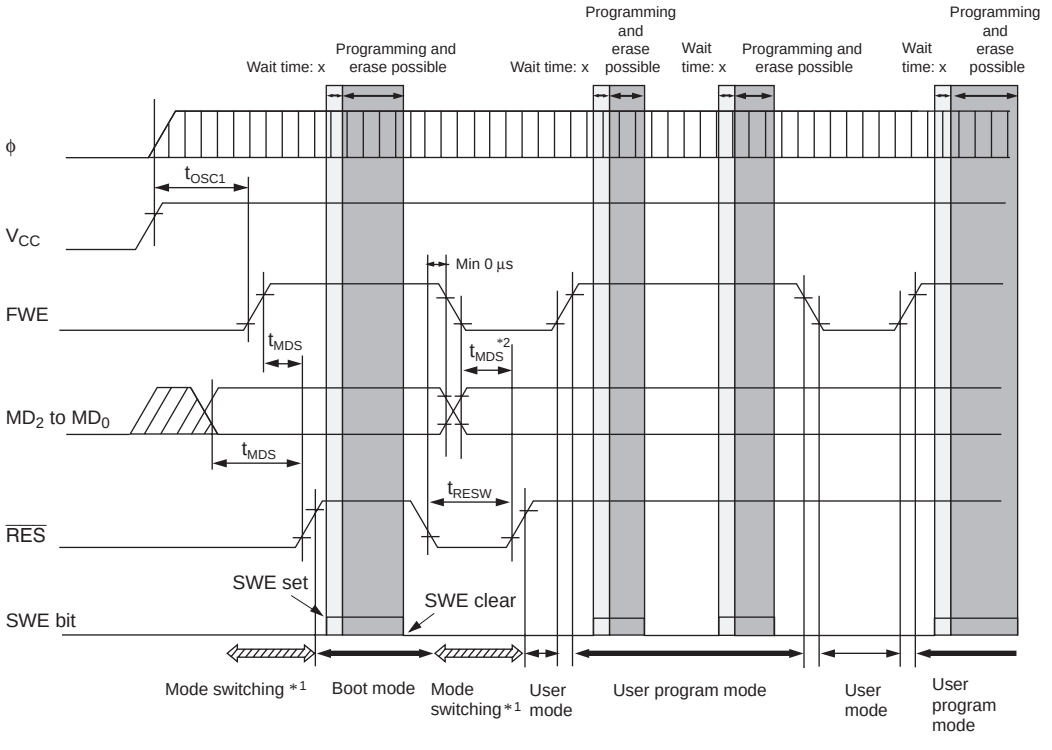



Flash memory access disabled period
(x: Wait time after SWE setting) *2

Flash memory reprogrammable period
(Flash memory program execution and data read, other than verify, are disabled.)

Notes: 1. Always fix the level by pulling down or pulling up the mode pins ($MD_2$ to $MD_0$) up to powering off, except for mode switching.
2. See section 20.1.6 Flash Memory Characteristics.
3. Mode programming setup time $t_{MDS}$.

**Figure 17.37 Powering On/Off Timing (User Program Mode)**




Flash memory access disabled period
(x: Wait time after SWE setting) *3

Flash memory reprogammable period
(Flash memory program execution and data read, other than verify, are disabled.)

Notes: 1. In transition to the boot mode and transition from the boot mode to another mode, mode switching via $\overline{RES}$ input is necessary.
During this switching period (period during which a low level is input to the $\overline{RES}$ pin), the state of the address dual port and bus control output signals ($\overline{AS}$, $\overline{RD}$, $\overline{HUR}$, $\overline{LWR}$) changes.
Therefore, do not use these pins as output signals during this switching period.
2. When making a transition from the boot mode to another mode, the mode programming setup time $t_{MDS}$ relative to the $\overline{RES}$ clear timing is necessary.
3. See section 20.1.6 Flash Memory Characteristics.

**Figure 17.38 Mode Transition Timing**
**(Example: Boot Mode → User Mode ↔ User Program Mode**

# Section 18 Clock Pulse Generator

## 18.1 Overview

The H8S/2345 Series has a built-in clock pulse generator (CPG) that generates the system clock (ø), the bus master clock, and internal clocks.

The clock pulse generator consists of an oscillator circuit, a duty adjustment circuit, a medium-speed clock divider, and a bus master clock selection circuit.

### 18.1.1 Block Diagram

Figure 18.1 shows a block diagram of the clock pulse generator.

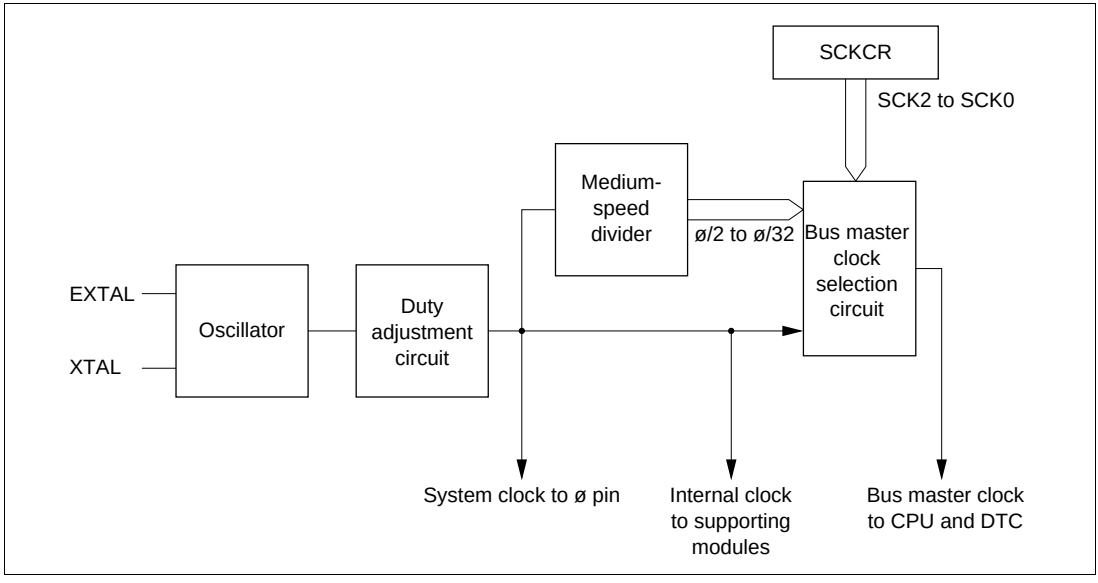



**Figure 18.1 Block Diagram of Clock Pulse Generator**

### 18.1.2 Register Configuration

The clock pulse generator is controlled by SCKCR. Table 18.1 shows the register configuration.

**Table 18.1 Clock Pulse Generator Register**

| Name | Abbreviation | R/W | Initial Value | Address* |
|---|---|---|---|---|
| System clock control register | SCKCR | R/W | H'00 | H'FF3A |

Note: * Lower 16 bits of the address.

## 18.2 Register Descriptions

### 18.2.1 System Clock Control Register (SCKCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PSTOP | — | — | — | — | SCK2 | SCK1 | SCK0 |
| Initial value: | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | — | — | — | R/W | R/W | R/W |

SCKCR is an 8-bit readable/writable register that performs ø clock output control and medium-speed mode control.

SCKCR is initialized to H'00 by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 7—ø Clock Output Disable (PSTOP):** Controls ø output.

| | Description | | | |
|---|---|---|---|---|
| **Bit 7** | | | | |
| **PSTOP** | **Normal Operation** | **Sleep Mode** | **Software Standby Mode** | **Hardware Standby Mode** |
| 0 | ø output (initial value) | ø output | Fixed high | High impedance |
| 1 | Fixed high | Fixed high | Fixed high | High impedance |

**Bit 6—Reserved:** This bit can be read or written to, but only 0 should be written.

**Bits 5 to 3—Reserved:** Read-only bits, always read as 0.

**Bits 2 to 0—System Clock Select 2 to 0 (SCK2 to SCK0):** These bits select the clock for the bus master.

| **Bit 2** | **Bit 1** | **Bit 0** | | |
|---|---|---|---|---|
| **SCK2** | **SCK1** | **SCK0** | **Description** | |
| 0 | 0 | 0 | Bus master is in high-speed mode | (Initial value) |
| | | 1 | Medium-speed clock is ø/2 | |
| | 1 | 0 | Medium-speed clock is ø/4 | |
| | | 1 | Medium-speed clock is ø/8 | |
| 1 | 0 | 0 | Medium-speed clock is ø/16 | |
| | | 1 | Medium-speed clock is ø/32 | |
| | 1 | — | — | |

## 18.3 Oscillator

Clock pulses can be supplied by connecting a crystal resonator, or by input of an external clock.

### 18.3.1 Connecting a Crystal Resonator

**Circuit Configuration:** A crystal resonator can be connected as shown in the example in figure 18.2. Select the damping resistance $R_d$ according to table 18.2. An AT-cut parallel-resonance crystal should be used.

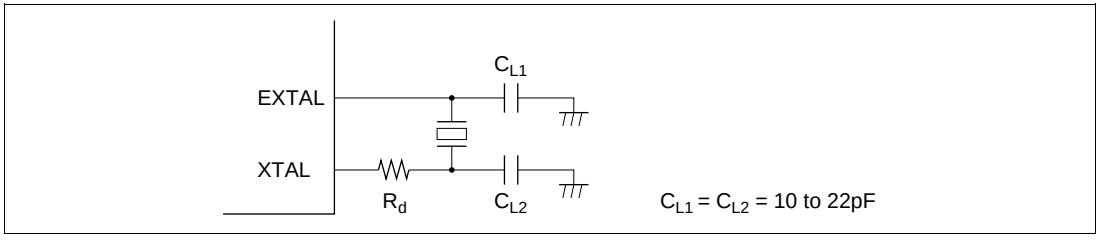



**Figure 18.2 Connection of Crystal Resonator (Example)**

**Table 18.2 Damping Resistance Value**

| Frequency (MHz) | 2 | 4 | 8 | 12 | 16 | 20 |
|---|---|---|---|---|---|---|
| $R_d$ (Ω) | 1k | 500 | 200 | 0 | 0 | 0 |

**Crystal Resonator:** Figure 18.3 shows the equivalent circuit of the crystal resonator. Use a crystal resonator that has the characteristics shown in table 18.3 and the same resonance frequency as the system clock (ø).

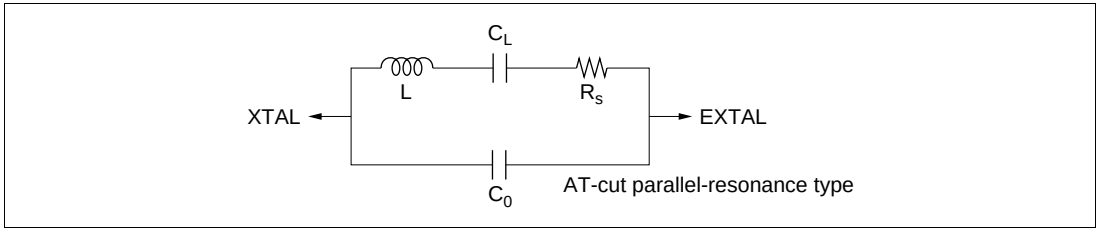



**Figure 18.3 Crystal Resonator Equivalent Circuit**

**Table 18.3 Crystal Resonator Parameters**

| Frequency (MHz) | 2 | 4 | 8 | 12 | 16 | 20 |
|---|---|---|---|---|---|---|
| $R_S$ max (Ω) | 500 | 120 | 80 | 60 | 50 | 40 |
| $C_0$ max (pF) | 7 | 7 | 7 | 7 | 7 | 7 |

**Note on Board Design:** When a crystal resonator is connected, the following points should be noted:

Other signal lines should be routed away from the oscillator circuit to prevent induction from interfering with correct oscillation. See figure 18.4.

When designing the board, place the crystal resonator and its load capacitors as close as possible to the XTAL and EXTAL pins.

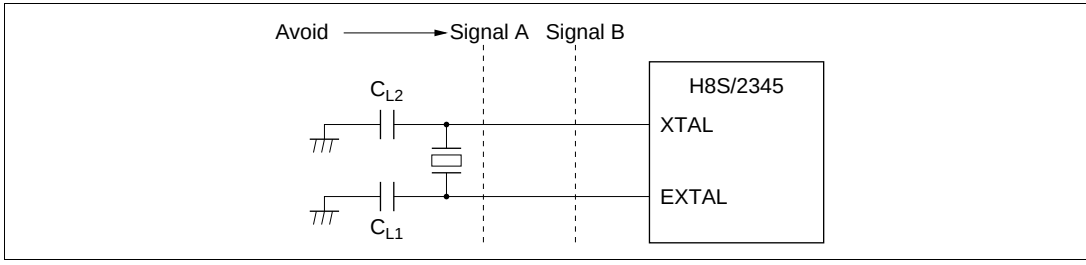



**Figure 18.4 Example of Incorrect Board Design**

### 18.3.2 External Clock Input

**Circuit Configuration:** An external clock signal can be input as shown in the examples in figure 18.5. If the XTAL pin is left open, make sure that stray capacitance is no more than 10 pF.

In example (b), make sure that the external clock is held high in standby mode.

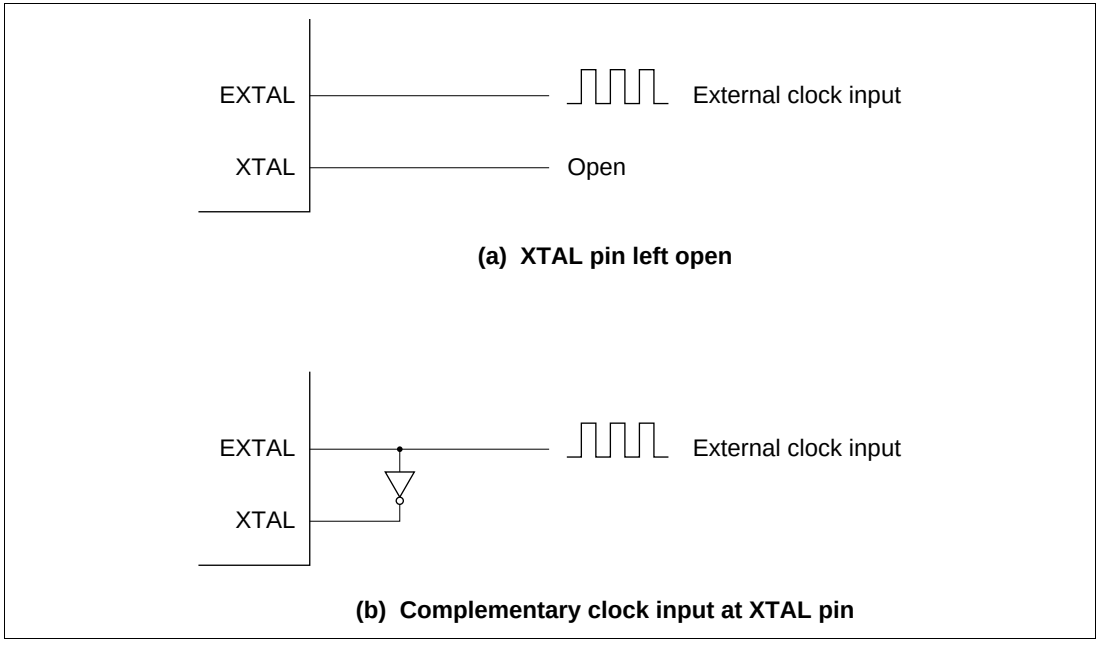



**Figure 18.5 External Clock Input (Examples)**

**External Clock:** The external clock signal should have the same frequency as the system clock (ø).

Table 18.4 and figure 18.6 show the input conditions for the external clock.

**Table 18.4 External Clock Input Conditions**

| Item | Symbol | $V_{cc}$ = 2.7 V* to 5.5 V Min | Max | $V_{cc}$ = 5.0 V ± 10% Min | Max | Unit | Test Conditions | |
|---|---|---|---|---|---|---|---|---|
| External clock input low pulse width | $t_{EXL}$ | 40 | — | 20 | — | ns | Figure 18.6 | |
| External clock input high pulse width | $t_{EXH}$ | 40 | — | 20 | — | ns | | |
| External clock rise time | $t_{EXr}$ | — | 10 | — | 5 | ns | | |
| External clock fall time | $t_{EXf}$ | — | 10 | — | 5 | ns | | |
| Clock low pulse width level | $t_{CL}$ | 0.4 | 0.6 | 0.4 | 0.6 | $t_{cyc}$ | ø ≥ 5 MHz | Figure 20.4 |
| | | 80 | — | 80 | — | ns | ø < 5 MHz | |
| Clock high pulse width level | $t_{CH}$ | 0.4 | 0.6 | 0.4 | 0.6 | $t_{cyc}$ | ø ≥ 5 MHz | |
| | | 80 | — | 80 | — | ns | ø < 5 MHz | |

Note: * ZTAT, mask ROM, and ROMless versions only.

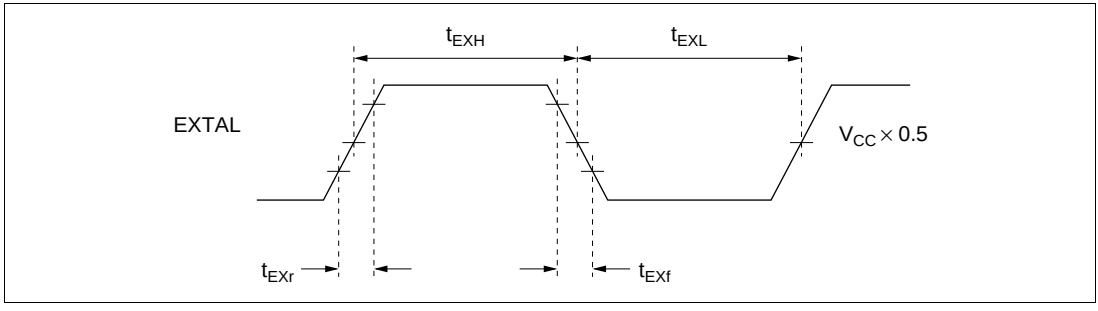



**Figure 18.6 External Clock Input Timing**

## 18.4 Duty Adjustment Circuit

When the oscillator frequency is 5 MHz or higher, the duty adjustment circuit adjusts the duty cycle of the clock signal from the oscillator to generate the system clock (ø).

## 18.5 Medium-Speed Clock Divider

The medium-speed clock divider divides the system clock to generate ø/2, ø/4, ø/8, ø/16, and ø/32.

## 18.6 Bus Master Clock Selection Circuit

The bus master clock selection circuit selects the system clock (ø) or one of the medium-speed clocks (ø/2, ø/4, or ø/8, ø/16, and ø/32) to be supplied to the bus master, according to the settings of the SCK2 to SCK0 bits in SCKCR.

# Section 19 Power-Down Modes

## 19.1 Overview

In addition to the normal program execution state, the H8S/2345 Series has five power-down modes in which operation of the CPU and oscillator is halted and power dissipation is reduced. Low-power operation can be achieved by individually controlling the CPU, on-chip supporting modules, and so on.

The H8S/2345 Series operating modes are as follows:

(1) High-speed mode
(2) Medium-speed mode
(3) Sleep mode
(4) Module stop mode
(5) Software standby mode
(6) Hardware standby mode

Of these, (2) to (6) are power-down modes. Sleep mode is a CPU mode, medium-speed mode is a CPU and bus master mode, and module stop mode is an on-chip supporting module mode (including bus masters other than the CPU). A combination of these modes can be set.

After a reset, the H8S/2345 Series is in high-speed mode.

Table 19.1 shows the conditions for transition to the various modes, the status of the CPU, on-chip supporting modules, etc., and the method of clearing each mode.

**Table 19.1 Operating Modes**

| Operating Mode | Transition Condition | Clearing Condition | Oscillator | CPU | | Modules | | I/O Ports |
|---|---|---|---|---|---|---|---|---|
| | | | | | Registers | | Registers | |
| High speed mode | Control register | | Functions | High speed | Functions | High speed | Functions | High speed |
| Medium-speed mode | Control register | | Functions | Medium speed | Functions | High/ medium speed *1 | Functions | High speed |
| Sleep mode | Instruction | Interrupt | Functions | Halted | Retained | High speed | Functions | High speed |
| Module stop mode | Control register | | Functions | High/ medium speed | Functions | Halted | Retained/ reset *2 | Retained |
| Software standby mode | Instruction | External interrupt | Halted | Halted | Retained | Halted | Retained/ reset *2 | Retained |
| Hardware standby mode | Pin | | Halted | Halted | Undefined | Halted | Reset | High impedance |

Notes: 1. The bus master operates on the medium-speed clock, and other on-chip supporting modules on the high-speed clock.
2. The SCI and A/D converter are reset, and other on-chip supporting modules retain their state.

### 19.1.1 Register Configuration

Power-down modes are controlled by the SBYCR, SCKCR, and MSTPCR registers. Table 19.2 summarizes these registers.

**Table 19.2 Power-Down Mode Registers**

| Name | Abbreviation | R/W | Initial Value | Address* |
|---|---|---|---|---|
| Standby control register | SBYCR | R/W | H'08 | H'FF38 |
| System clock control register | SCKCR | R/W | H'00 | H'FF3A |
| Module stop control register H | MSTPCRH | R/W | H'3F | H'FF3C |
| Module stop control register L | MSTPCRL | R/W | H'FF | H'FF3D |

Note: * Lower 16 bits of the address.

## 19.2 Register Descriptions

### 19.2.1 Standby Control Register (SBYCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | SSBY | STS2 | STS1 | STS0 | OPE | — | — | — |
| Initial value | : | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | — | — | R/W |

SBYCR is an 8-bit readable/writable register that performs software standby mode control.

SBYCR is initialized to H'08 by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 7—Software Standby (SSBY):** Specifies a transition to software standby mode. Remains set to 1 when software standby mode is released by an external interrupt, and a transition is made to normal operation. The SSBY bit should be cleared by writing 0 to it.

**Bit 7**

| SSBY | Description | |
|---|---|---|
| 0 | Transition to sleep mode after execution of SLEEP instruction | (Initial value) |
| 1 | Transition to software standby mode after execution of SLEEP instruction | |

**Bits 6 to 4—Standby Timer Select 2 to 0 (STS2 to STS0):** These bits select the time the MCU waits for the clock to stabilize when software standby mode is cleared by an external interrupt. With crystal oscillation, refer to table 19.4 and make a selection according to the operating frequency so that the standby time is at least 8 ms (the oscillation stabilization time). With an external clock, any selection* can be made.

Note: * The 16-state standby time cannot be used in the F-ZTAT version; a standby time of 8192 states or longer should be used.

| Bit 6 | Bit 5 | Bit 4 | | |
|---|---|---|---|---|
| **STS2** | **STS1** | **STS0** | **Description** | |
| 0 | 0 | 0 | Standby time = 8192 states | (Initial value) |
| | | 1 | Standby time = 16384 states | |
| | 1 | 0 | Standby time = 32768 states | |
| | | 1 | Standby time = 65536 states | |
| 1 | 0 | 0 | Standby time = 131072 states | |
| | | 1 | Standby time = 262144 states | |
| | 1 | 0 | Reserved | |
| | | 1 | Standby time = 16 states* | |

Note: * Not used on the F-ZTAT version.

**Bit 3—Output Port Enable (OPE):** Specifies whether the output of the address bus and bus control signals ($\overline{CS0}$ to $\overline{CS3}$, $\overline{AS}$, $\overline{RD}$, $\overline{HWR}$, $\overline{LWR}$) is retained or set to the high-impedance state in software standby mode.

| Bit 3 | |
|---|---|
| **OPE** | **Description** |
| 0 | In software standby mode, address bus and bus control signals are high-impedance |
| 1 | In software standby mode, address bus and bus control signals retain output state (Initial value) |

**Bits 2 and 1—Reserved:** Read-only bits, always read as 0.

**Bit 0—Reserved:** This bit can be read or written to, but only 0 should be written.

### 19.2.2 System Clock Control Register (SCKCR)

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PSTOP | — | — | — | — | SCK2 | SCK1 | SCK0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | R/W | R/W | — | — | — | R/W | R/W | R/W |

SCKCR is an 8-bit readable/writable register that performs ø clock output control and medium-speed mode control.

SCKCR is initialized to H'00 by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 7—ø Clock Output Disable (PSTOP):** Controls ø output.

| Bit 7 | Description | | | |
|---|---|---|---|---|
| PSTOP | Normal Operating Mode | Sleep Mode | Software Standby Mode | Hardware Standby Mode |
| 0 | ø output (initial value) | ø output | Fixed high | High impedance |
| 1 | Fixed high | Fixed high | Fixed high | High impedance |

**Bits 6—Reserved:** This bit can be read or written to, but only 0 should be written.

**Bits 5 to 3—Reserved:** Read-only bits, always read as 0.

**Bits 2 to 0—System Clock Select (SCK2 to SCK0):** These bits select the clock for the bus master.

| Bit 2 | Bit 1 | Bit 0 | | |
|---|---|---|---|---|
| SCK2 | SCK1 | SCK0 | Description | |
| 0 | 0 | 0 | Bus master in high-speed mode | (Initial value) |
| | | 1 | Medium-speed clock is ø/2 | |
| | 1 | 0 | Medium-speed clock is ø/4 | |
| | | 1 | Medium-speed clock is ø/8 | |
| 1 | 0 | 0 | Medium-speed clock is ø/16 | |
| | | 1 | Medium-speed clock is ø/32 | |
| | 1 | — | — | |

### 19.2.3 Module Stop Control Register (MSTPCR)

| | | MSTPCRH | | | | | | | | MSTPCRL | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Initial value | : | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bits 15 to 0—Module Stop (MSTP 15 to MSTP 0):** These bits specify module stop mode. See table 19.3 for the method of selecting on-chip supporting modules.

**Bits 15 to 0**

| MSTP15 to MSTP0 | Description |
|---|---|
| 0 | Module stop mode cleared |
| 1 | Module stop mode set |

## 19.3 Medium-Speed Mode

When the SCK2 to SCK0 bits in SCKCR are set to 1, the operating mode changes to medium-speed mode as soon as the current bus cycle ends. In medium-speed mode, the CPU operates on the operating clock (ø/2, ø/4, ø/8, ø/16, or ø/32) specified by the SCK2 to SCK0 bits. The bus masters other than the CPU (the DTC) also operate in medium-speed mode. On-chip supporting modules other than the bus masters always operate on the high-speed clock (ø).

In medium-speed mode, a bus access is executed in the specified number of states with respect to the bus master operating clock. For example, if ø/4 is selected as the operating clock, on-chip memory is accessed in 4 states, and internal I/O registers in 8 states.

Medium-speed mode is cleared by clearing all of bits SCK2 to SCK0 to 0. A transition is made to high-speed mode and medium-speed mode is cleared at the end of the current bus cycle.

If a SLEEP instruction is executed when the SSBY bit in SBYCR is cleared to 0, a transition is made to sleep mode. When sleep mode is cleared by an interrupt, medium-speed mode is restored.

If a SLEEP instruction is executed when the SSBY bit in SBYCR is set to 1, a transition is made to software standby mode. When software standby mode is cleared by an external interrupt, medium-speed mode is restored.

When the $\overline{\text{RES}}$ pin is driven low, a transition is made to the reset state, and medium-speed mode is cleared. The same applies in the case of a reset caused by overflow of the watchdog timer.

When the $\overline{\text{STBY}}$ pin is driven low, a transition is made to hardware standby mode.

Figure 19.1 shows the timing for transition to and clearance of medium-speed mode.

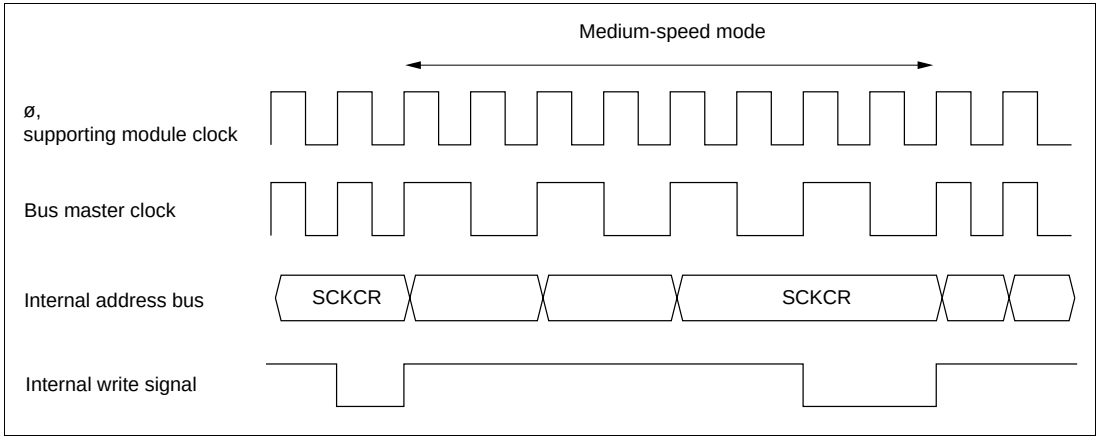



**Figure 19.1 Medium-Speed Mode Transition and Clearance Timing**

## 19.4 Sleep Mode

If a SLEEP instruction is executed when the SSBY bit in SBYCR is cleared to 0, the CPU enters sleep mode. In sleep mode, CPU operation stops but the contents of the CPU's internal registers are retained. Other supporting modules do not stop.

Sleep mode is cleared by a reset or any interrupt, and the CPU returns to the normal program execution state via the exception handling state. Sleep mode is not cleared if interrupts are disabled, or if interrupts other than NMI are masked by the CPU.

When the $\overline{\text{STBY}}$ pin is driven low, a transition is made to hardware standby mode.

## 19.5 Module Stop Mode

### 19.5.1 Module Stop Mode

Module stop mode can be set for individual on-chip supporting modules.

When the corresponding MSTP bit in MSTPCR is set to 1, module operation stops at the end of the bus cycle and a transition is made to module stop mode. The CPU continues operating independently.

Table 19.3 shows MSTP bits and the corresponding on-chip supporting modules.

When the corresponding MSTP bit is cleared to 0, module stop mode is cleared and the module starts operating at the end of the bus cycle. In module stop mode, the internal states of modules other than the SCI and A/D converter are retained.

After reset clearance, all modules other than DTC are in module stop mode.

When an on-chip supporting module is in module stop mode, read/write access to its registers is disabled.

If a transition is made to sleep mode when all modules are stopped (MSTPCR = H'FFFF), or modules other than the 8-bit timers are stopped (MSTPCR = H'EFFF), operation of the bus controller and I/O ports is also halted, enabling current dissipation to be further reduced.

**Table 19.3 MSTP Bits and Corresponding On-Chip Supporting Modules**

| Register | Bit | Module |
|---|---|---|
| MSTPCRH | MSTP15 | — |
| | MSTP14 | Data transfer controller (DTC) |
| | MSTP13 | 16-bit timer pulse unit (TPU) |
| | MSTP12 | 8-bit timer |
| | MSTP11 | — |
| | MSTP10 | D/A converter |
| | MSTP9 | A/D converter |
| | MSTP8 | — |
| MSTPCRL | MSTP7 | — |
| | MSTP6 | Serial communication interface (SCI) channel 1 |
| | MSTP5 | Serial communication interface (SCI) channel 0 |
| | MSTP4 | — |
| | MSTP3 | — |
| | MSTP2 | — |
| | MSTP1 | — |
| | MSTP0 | — |

Note: Bits 15, 11, 8, 7, and 4 to 0 can be read or written to, but do not affect operation.

### 19.5.2 Usage Notes

**DTC Module Stop:** Depending on the operating status of the DTC, the MSTP14 bit may not be set to 1. Setting of the DTC module stop mode should be carried out only when the respective module is not activated.

For details, refer to section 7, Data Transfer Controller (DTC).

**On-Chip Supporting Module Interrupt:** Relevant interrupt operations cannot be performed in module stop mode. Consequently, if module stop mode is entered when an interrupt has been requested, it will not be possible to clear the CPU interrupt source or the DTC activation source. Interrupts should therefore be disabled before entering module stop mode.

**Writing to MSTPCR:** MSTPCR should only be written to by the CPU.

## 19.6 Software Standby Mode

### 19.6.1 Software Standby Mode

If a SLEEP instruction is executed when the SSBY bit in SBYCR is set to 1, software standby mode is entered. In this mode, the CPU, on-chip supporting modules, and oscillator all stop. However, the contents of the CPU's internal registers, RAM data, and the states of on-chip supporting modules other than the SCI and A/D converter, and I/O ports, are retained. Whether the address bus and bus control signals are placed in the high-impedance state or retain the output state can be specified by the OPE bit in SBYCR.

In this mode the oscillator stops, and therefore power dissipation is significantly reduced.

### 19.6.2 Clearing Software Standby Mode

Software standby mode is cleared by an external interrupt (NMI pin, or pins $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ2}}$), or by means of the $\overline{\text{RES}}$ pin or $\overline{\text{STBY}}$ pin.

- Clearing with an interrupt

  When an NMI or IRQ0 to IRQ2 interrupt request signal is input, clock oscillation starts, and after the elapse of the time set in bits STS2 to STS0 in SYSCR, stable clocks are supplied to the entire H8S/2345 Series chip, software standby mode is cleared, and interrupt exception handling is started.

  When clearing software standby mode with an IRQ0 to IRQ2 interrupt, set the corresponding enable bit to 1 and ensure that no interrupt with a higher priority than interrupts IRQ0 to IRQ2 is generated. Software standby mode cannot be cleared if the interrupt has been masked on the CPU side or has been designated as a DTC activation source.

- Clearing with the $\overline{\text{RES}}$ pin

  When the $\overline{\text{RES}}$ pin is driven low, clock oscillation is started. At the same time as clock oscillation starts, clocks are supplied to the entire H8S/2345 Series chip. Note that the $\overline{\text{RES}}$ pin must be held low until clock oscillation stabilizes. When the $\overline{\text{RES}}$ pin goes high, the CPU begins reset exception handling.

- Clearing with the $\overline{\text{STBY}}$ pin

  When the $\overline{\text{STBY}}$ pin is driven low, a transition is made to hardware standby mode.

### 19.6.3 Setting Oscillation Stabilization Time after Clearing Software Standby Mode

Bits STS2 to STS0 in SBYCR should be set as described below.

**Using a Crystal Oscillator:** Set bits STS2 to STS0 so that the standby time is at least 8 ms (the oscillation stabilization time).

Table 19.4 shows the standby times for different operating frequencies and settings of bits STS2 to STS0.

**Table 19.4 Oscillation Stabilization Time Settings**

| STS2 | STS1 | STS0 | Standby Time | 20 MHz | 16 MHz | 12 MHz | 10 MHz | 8 MHz | 6 MHz | 4 MHz | 2 MHz | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 8192 states | 0.41 | 0.51 | 0.68 | 0.8 | 1.0 | 1.3 | 2.0 | 4.1 | ms |
| | | 1 | 16384 states | 0.82 | 1.0 | 1.3 | 1.6 | 2.0 | 2.7 | 4.1 | [8.2] | |
| | 1 | 0 | 32768 states | 1.6 | 2.0 | 2.7 | 3.3 | 4.1 | 5.5 | [8.2] | 16.4 | |
| | | 1 | 65536 states | 3.3 | 4.1 | 5.5 | 6.6 | [8.2] | [10.9] | 16.4 | 32.8 | |
| 1 | 0 | 0 | 131072 states | 6.6 | [8.2] | [10.9] | [13.1] | 16.4 | 21.8 | 32.8 | 65.5 | |
| | | 1 | 262144 states | [13.1] | 16.4 | 21.8 | 26.2 | 32.8 | 43.6 | 65.6 | 131.2 | |
| | 1 | 0 | Reserved | — | — | — | — | — | — | — | — | — |
| | | 1 | 16 states* | 0.8 | 1.0 | 1.3 | 1.6 | 2.0 | 2.7 | 4.0 | 8.0 | µs |

[ ]: Recommended time setting

Note: * Not used on the F-ZTAT version.

**Using an External Clock:** Any value can be set. Normally, use of the minimum time is recommended.

Note: * The 16-state standby time cannot be used in the F-ZTAT version; a standby time of 8192 states or longer should be used.

### 19.6.4 Software Standby Mode Application Example

Figure 19.2 shows an example in which a transition is made to software standby mode at the falling edge on the NMI pin, and software standby mode is cleared at the rising edge on the NMI pin.

In this example, an NMI interrupt is accepted with the NMIEG bit in SYSCR cleared to 0 (falling edge specification), then the NMIEG bit is set to 1 (rising edge specification), the SSBY bit is set to 1, and a SLEEP instruction is executed, causing a transition to software standby mode.

Software standby mode is then cleared at the rising edge on the NMI pin.

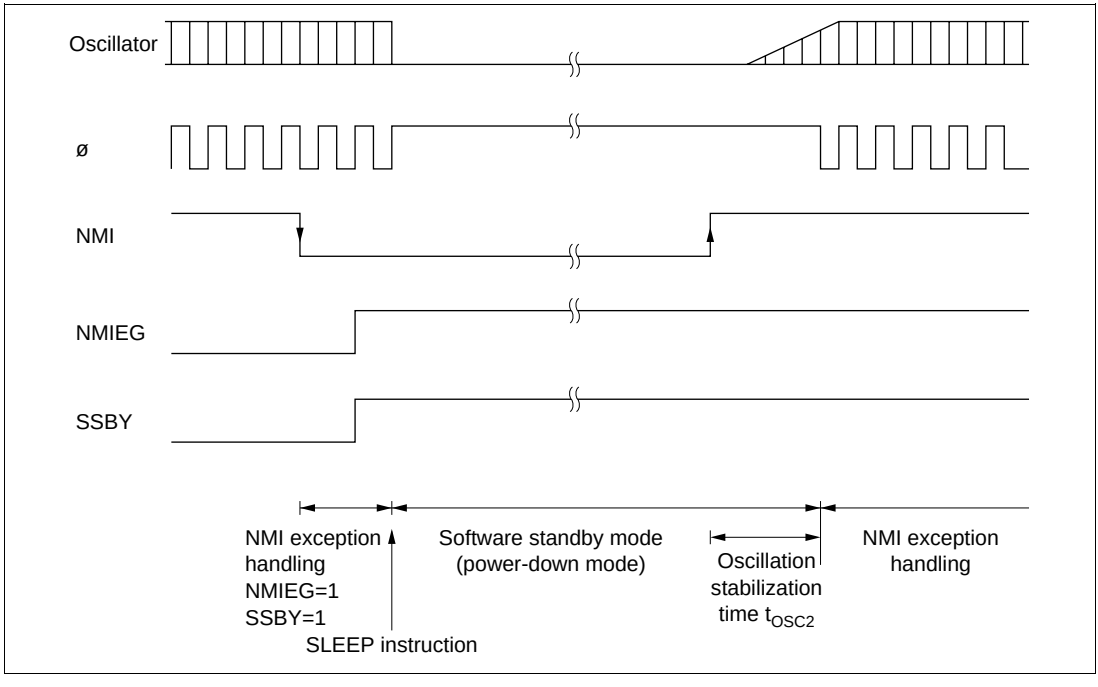



**Figure 19.2 Software Standby Mode Application Example**

### 19.6.5 Usage Notes

**I/O Port Status:** In software standby mode, I/O port states are retained. If the OPE bit is set to 1, the address bus and bus control signal output is also retained. Therefore, there is no reduction in current dissipation for the output current when a high-level signal is output.

**Current Dissipation during Oscillation Stabilization Wait Period:** Current dissipation increases during the oscillation stabilization wait period.

## 19.7 Hardware Standby Mode

### 19.7.1 Hardware Standby Mode

When the $\overline{\text{STBY}}$ pin is driven low, a transition is made to hardware standby mode from any mode.

In hardware standby mode, all functions enter the reset state and stop operation, resulting in a significant reduction in power dissipation. As long as the prescribed voltage is supplied, on-chip RAM data is retained. I/O ports are set to the high-impedance state.

In order to retain on-chip RAM data, the RAME bit in SYSCR should be cleared to 0 before driving the $\overline{\text{STBY}}$ pin low.

Do not change the state of the mode pins ($MD_2$ to $MD_0$) while the H8S/2345 Series is in hardware standby mode.

Hardware standby mode is cleared by means of the $\overline{\text{STBY}}$ pin and the $\overline{\text{RES}}$ pin. When the $\overline{\text{STBY}}$ pin is driven high while the $\overline{\text{RES}}$ pin is low, the reset state is set and clock oscillation is started. Ensure that the $\overline{\text{RES}}$ pin is held low until the clock oscillator stabilizes (at least 8 ms—the oscillation stabilization time—when using a crystal oscillator). When the $\overline{\text{RES}}$ pin is subsequently driven high, a transition is made to the program execution state via the reset exception handling state.

### 19.7.2 Hardware Standby Mode Timing

Figure 19.3 shows an example of hardware standby mode timing.

When the $\overline{\text{STBY}}$ pin is driven low after the $\overline{\text{RES}}$ pin has been driven low, a transition is made to hardware standby mode. Hardware standby mode is cleared by driving the $\overline{\text{STBY}}$ pin high, waiting for the oscillation stabilization time, then changing the $\overline{\text{RES}}$ pin from low to high.

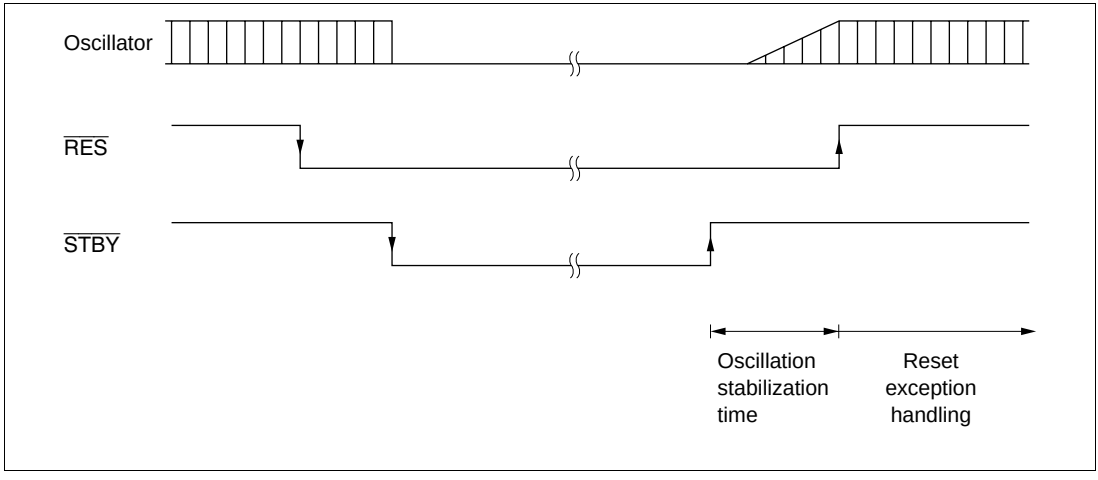



**Figure 19.3 Hardware Standby Mode Timing (Example)**

## 19.8 ø Clock Output Disabling Function

Output of the ø clock can be controlled by means of the PSTOP bit in SCKCR, and DDR for the corresponding port. When the PSTOP bit is set to 1, the ø clock stops at the end of the bus cycle, and ø output goes high. ø clock output is enabled when the PSTOP bit is cleared to 0. When DDR for the corresponding port is cleared to 0, ø clock output is disabled and input port mode is set. Table 19.5 shows the state of the ø pin in each processing state.

**Table 19.5 ø Pin State in Each Processing State**

| DDR | 0 | 1 | |
|---|---|---|---|
| PSTOP | — | 0 | 1 |
| Hardware standby mode | High impedance | | |
| Software standby mode | High impedance | Fixed high | |
| Sleep mode | High impedance | ø output | Fixed high |
| Normal operating state | High impedance | ø output | Fixed high |

## 20.1 Electrical Characteristics of F-ZTAT Version

### 20.1.1 Absolute Maximum Ratings

Table 20.1 lists the absolute maximum ratings.

**Table 20.1 Absolute Maximum Ratings**

| Item | Symbol | Value | Unit |
|---|---|---|---|
| Power supply voltage | $V_{CC}$ | –0.3 to +7.0 | V |
| Input voltage (FWE)*1 | $V_{in}$ | –0.3 to $V_{CC}$ +0.3 | V |
| Input voltage (except port 4) | $V_{in}$ | –0.3 to $V_{CC}$ +0.3 | V |
| Input voltage (port 4) | $V_{in}$ | –0.3 to $AV_{CC}$ +0.3 | V |
| Reference voltage | $V_{ref}$ | –0.3 to $AV_{CC}$ +0.3 | V |
| Analog power supply voltage | $AV_{CC}$ | –0.3 to +7.0 | V |
| Analog input voltage | $V_{AN}$ | –0.3 to $AV_{CC}$ +0.3 | V |
| Operating temperature | $T_{opr}$ | Regular specifications: –20 to +75*2 | °C |
| | | Wide-range specifications: –40 to +85*2 | °C |
| Storage temperature | $T_{stg}$ | –55 to +125 | °C |

Caution: Permanent damage to the chip may result if absolute maximum rating are exceeded.

Notes: 1. Never apply 12 V to any of the pins. Doing so could permanently damage the LSI.
2. The operating temperature range for flash memory programming/erase operations is $T_a$ = 0 to +75°C (regular specifications), $T_a$ = 0 to +85°C (wide-range specifications).

### 20.1.2 DC Characteristics

Table 20.2 lists the DC characteristics. Table 20.3 lists the permissible output currents.

**Table 20.2 DC Characteristics**

Conditions: $V_{CC} = AV_{CC} = 5.0\ V \pm 10\%$, $V_{ref} = 4.5\ V$ to $AV_{CC}$, $V_{SS} = AV_{SS} = 0\ V$*[1], $T_a = -20$ to $+75°C$ (regular specifications), $T_a = -40$ to $+85°C$ (wide-range specifications)

| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|
| Schmitt trigger input voltage | Port 2, $\overline{IRQ0}$ to $\overline{IRQ7}$ | $V_T^-$ | 1.0 | — | — | V | |
| | | $V_T^+$ | — | — | $V_{CC} \times 0.7$ | V | |
| | | $V_T^+ - V_T^-$ | 0.4 | — | — | V | |
| Input high voltage | $\overline{RES}$, $\overline{STBY}$, NMI, $MD_2$ to $MD_0$, FWE | $V_{IH}$ | $V_{CC} - 0.7$ | — | $V_{CC} + 0.3$ | V | |
| | EXTAL | | $V_{CC} \times 0.7$ | — | $V_{CC} + 0.3$ | V | |
| | Port 1, 3, A to G | | 2.0 | — | $V_{CC} + 0.3$ | V | |
| | Port4 | | 2.0 | — | $AV_{CC} + 0.3$ | V | |
| Input low voltage | $\overline{RES}$, $\overline{STBY}$, $MD_2$ to $MD_0$, FWE | $V_{IL}$ | –0.3 | — | 0.5 | V | |
| | NMI, EXTAL, Port 1, 3, 4, A to G | | –0.3 | — | 0.8 | V | |
| Output high voltage | All output pins | $V_{OH}$ | $V_{CC} - 0.5$ | — | — | V | $I_{OH} = -200\ \mu A$ |
| | | | 3.5 | — | — | V | $I_{OH} = -1\ mA$ |
| Output low voltage | All output pins | $V_{OL}$ | — | — | 0.4 | V | $I_{OL} = 1.6\ mA$ |
| | Port 1, A to C | | — | — | 1.0 | V | $I_{OL} = 10\ mA$ |
| Input leakage current | $\overline{RES}$ | $\lvert I_{in} \rvert$ | — | — | 10.0 | µA | $V_{in}$ = 0.5 to $V_{CC} - 0.5$ V |
| | $\overline{STBY}$, NMI, $MD_2$ to $MD_0$, FWE | | — | — | 1.0 | µA | |
| | Port 4 | | — | — | 1.0 | µA | $V_{in}$ = 0.5 to $AV_{CC} - 0.5$ V |

Note: 1. If the A/D and D/A converters are not used, do not leave the $AV_{CC}$, $AV_{SS}$, and $V_{ref}$ pins open.
Connect $AV_{CC}$ and $V_{ref}$ to $V_{CC}$, and connect $AV_{SS}$ to $V_{SS}$.

**Table 20.2 DC Characteristics (cont)**

Conditions: $V_{CC} = AV_{CC} = 5.0\ V \pm 10\%$, $V_{ref} = 4.5\ V$ to $AV_{CC}$, $V_{SS} = AV_{SS} = 0\ V$*[1], $T_a = -20$ to $+75°C$ (regular specifications), $T_a = -40$ to $+85°C$ (wide-range specifications)

| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|
| Three-state leakage current (off state) | Port 1 to 3, A to G | $\lvert I_{TSI} \rvert$ | — | — | 1.0 | µA | $V_{in}$ = 0.5 to $V_{CC}$ – 0.5 V |
| MOS input pull-up current | Port A to E | $-I_P$ | 50 | — | 300 | µA | $V_{in}$ = 0 V |
| Input capacitance | $\overline{RES}$ | $C_{in}$ | — | — | 80 | pF | $V_{in}$ = 0 V, f = 1 MHz, $T_a$ = 25°C |
| | NMI | | — | — | 50 | pF | |
| | All input pins except $\overline{RES}$ and NMI | | — | — | 15 | pF | |
| Current dissipation*[2] | Normal operation | $I_{CC}$*[4] | — | 60 (5.0 V) | 89 | mA | f = 20 MHz |
| | Sleep mode | | — | 40 (5.0 V) | 73 | mA | f = 20 MHz |
| | Standby mode*[3] | | — | 0.01 | 5.0 | µA | $T_a \le 50°C$ |
| | | | — | — | 20 | | $50°C < T_a$ |
| | During flash memory programming/ erase | | — | 70 (5.0 V) | 89 | mA | $0°C \le T_a \le 75°C$, f = 20 MHz |
| Analog power supply current | During A/D and D/A conversion | $AI_{CC}$ | — | 0.8 (5.0 V) | 2.0 | mA | |
| | Idle | | — | 0.01 | 5.0 | µA | |
| Reference current | During A/D and D/A conversion | $AI_{CC}$ | — | 1.9 (5.0 V) | 3.0 | mA | $V_{ref}$ = 5.0 V |
| | Idle | | — | 0.01 | 5.0 | µA | |
| RAM standby voltage | | $V_{RAM}$ | 2.0 | — | — | V | |

Notes: 1. If the A/D and D/A converters are not used, do not leave the $AV_{CC}$, $AV_{SS}$, and $V_{ref}$ pins open.
Connect $AV_{CC}$ and $V_{ref}$ to $V_{CC}$, and connect $AV_{SS}$ to $V_{SS}$.

2. Current dissipation values are for $V_{IH}$ min = $V_{CC}$ –0.5 V and $V_{IL}$ max = 0.5V with all output pins unloaded and the on-chip pull-up transistors in the off state.

3. The values are for $V_{RAM} \le V_{CC} < 4.5V$, $V_{IH}$ min = $V_{CC} \times 0.9$, and $V_{IL}$ max = 0.3 V.

4. $I_{CC}$ depends on $V_{CC}$ and f as follows:
   $I_{CC}$ max = 1.0 (mA) + 0.80 (mA/(MHz $\times$ V)) $\times$ $V_{CC}$ $\times$ f [normal mode]
   $I_{CC}$ max = 1.0 (mA) + 0.65 (mA/(MHz $\times$ V)) $\times$ $V_{CC}$ $\times$ f [sleep mode]

Conditions: $V_{CC}$ = $AV_{CC}$ = 3.0 to 5.5 V, $V_{ref}$ = 3.0 V to $AV_{CC}$, $V_{SS}$ = $AV_{SS}$ = 0 V*[1], $T_a$ = –20 to +75°C (regular specifications), $T_a$ = –40 to +85°C (wide-range specifications)

| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|
| Schmitt trigger input voltage | Port 2, $\overline{IRQ0}$ to $\overline{IRQ7}$ | $V_T^-$ | $V_{CC} \times 0.2$ | — | — | V | |
| | | $V_T^+$ | — | — | $V_{CC} \times 0.7$ | V | |
| | | $V_T^+ - V_T^-$ | $V_{CC} \times 0.07$ | — | — | V | |
| Input high voltage | $\overline{RES}$, $\overline{STBY}$, NMI, $MD_2$ to $MD_0$, FWE | $V_{IH}$ | $V_{CC} \times 0.9$ | — | $V_{CC}$ +0.3 | V | |
| | EXTAL | | $V_{CC} \times 0.7$ | — | $V_{CC}$ +0.3 | V | |
| | Port 1, 3, A to G | | $V_{CC} \times 0.7$ | — | $V_{CC}$ +0.3 | V | |
| | Port 4 | | $V_{CC} \times 0.7$ | — | $AV_{CC}$ +0.3 | V | |
| Input low voltage | $\overline{RES}$, $\overline{STBY}$, $MD_2$ to $MD_0$, FWE | $V_{IL}$ | –0.3 | — | $V_{CC} \times 0.1$ | V | |
| | NMI, EXTAL, Port 1, 3 , 4, | | –0.3 | — | $V_{CC} \times 0.2$ | V | $V_{CC}$ < 4.0 V |
| | A to G | | | | 0.8 | | $V_{CC}$ = 4.0 to 5.5 V |
| Output high voltage | All output pins | $V_{OH}$ | $V_{CC}$ – 0.5 | — | — | V | $I_{OH}$ = –200 µA |
| | | | $V_{CC}$ – 1.0 | — | — | V | $I_{OH}$ = –1 mA |
| Output low voltage | All output pins | $V_{OL}$ | — | — | 0.4 | V | $I_{OL}$ = 1.6 mA |
| | Port 1, A to C | | — | — | 1.0 | V | $I_{OL}$ = 5 mA |
| Input leakage current | $\overline{RES}$ | $\lvert I_{in} \rvert$ | — | — | 10.0 | µA | $V_{in}$ = 0.5 to $V_{CC}$ – 0.5V |
| | $\overline{STBY}$, NMI, $MD_2$ to $MD_0$, FWE | | — | — | 1.0 | µA | |
| | Port 4 | | — | — | 1.0 | µA | $V_{in}$ = 0.5 to $AV_{CC}$ – 0.5V |

Note: 1. If the A/D and D/A converters are not used, do not leave the $AV_{CC}$, $AV_{SS}$, and $V_{ref}$ pins open.
Connect $AV_{CC}$ and $V_{ref}$ to $V_{CC}$, and connect $AV_{SS}$ to $V_{SS}$.

Conditions: $V_{CC}$ = $AV_{CC}$ = 3.0 to 3.6 V, $V_{ref}$ = 3.0 V to $AV_{CC}$, $V_{SS}$ = $AV_{SS}$ = 0 V*[1], $T_a$ = –20 to +75°C (regular specifications), $T_a$ = –40 to +85°C (wide-range specifications)

| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|
| Three-state leakage current (off state) | Port 1 to 3, A to G | $\lvert I_{TSI} \rvert$ | — | — | 1.0 | µA | $V_{in}$ = 0.5 to $V_{CC}$ –0.5 V |
| MOS input pull-up current | Port A to E | $-I_P$ | 10 | — | 300 | µA | $V_{CC}$ = 2.7 V to 5.5 V, $V_{in}$ = 0 V |
| Input capacitance | $\overline{RES}$ | $C_{in}$ | — | — | 80 | pF | $V_{in}$ = 0 V f = 1 MHz $T_a$ = 25°C |
| | NMI | | — | — | 50 | pF | |
| | All input pins except $\overline{RES}$ and NMI | | — | — | 15 | pF | |
| Current dissipation*[2] | Normal operation | $I_{CC}$*[4] | — | TBD (3.3 V) | TBD | mA | f = 10 MHz |
| | Sleep mode | | — | TBD (3.3 V) | TBD | mA | f = 10 MHz |
| | Standby mode*[3] | | — | 0.01 | 5.0 | µA | $T_a \leq 50°C$ |
| | | | — | — | 20 | | 50°C < $T_a$ |
| | During flash memory programming/ erase | | — | TBD (3.3 V) | TBD | mA | 0°C ≤ $T_a$ ≤ 75°C f = 10 MHz |
| Analog power supply current | During A/D and D/A conversion | $AI_{CC}$ | — | TBD (3.3 V) | TBD | mA | |
| | Idle | | — | 0.01 | 5.0 | µA | |
| Reference current | During A/D and D/A conversion | $AI_{CC}$ | — | TBD (3.3 V) | TBD | mA | $V_{ref}$ = 3.3 V |
| | Idle | | — | 0.01 | 5.0 | µA | |
| RAM standby voltage | | $V_{RAM}$ | 2.0 | — | — | V | |

Notes: 1. If the A/D and D/A converters are not used, do not leave the $AV_{CC}$, $AV_{SS}$, and Vref pins open.
Connect $AV_{CC}$ and $V_{ref}$ to $V_{CC}$, and connect $AV_{SS}$ to $V_{SS}$.
2. Current dissipation values are for $V_{IH}$ min = $V_{CC}$ –0.5 V and $V_{IL}$ max = 0.5V with all output pins unloaded and the on-chip pull-up transistors in the off state.
3. The values are for $V_{RAM} \leq V_{CC} < 2.7$ V, $V_{IH}$ min = $V_{CC} \times 0.9$, and $V_{IL}$ max = 0.3V.

4. $I_{CC}$ depends on $V_{CC}$ and f as follows:
   $I_{CC}$ max = TBD (mA) + TBD (mA/(MHz $\times$ V)) $\times$ $V_{CC}$ $\times$ f [normal mode]
   $I_{CC}$ max = TBD (mA) + TBD (mA/(MHz $\times$ V)) $\times$ $V_{CC}$ $\times$ f [sleep mode]

**Table 20.3 Permissible Output Currents**

Conditions: $V_{CC}$ = $AV_{CC}$ = 5.0 V ±10%, $V_{ref}$ = 4.5 to $AV_{CC}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $T_a$ = –20 to +75°C (regular specifications), $T_a$ = –40 to +85°C (wide-range specifications)

| Item | | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Permissible output low current (per pin) | Port 1, A to C | $I_{OL}$ | — | — | 10 | mA |
| | Other output pins | | — | — | 2.0 | mA |
| Permissible output low current (total) | Total of 28 pins including port 1 and A to C | $\Sigma I_{OL}$ | — | — | 80 | mA |
| | Total of all output pins, including the above | | — | — | 120 | mA |
| Permissible output high current (per pin) | All output pins | $-I_{OH}$ | — | — | 2.0 | mA |
| Permissible output high current (total) | Total of all output pins | $\Sigma -I_{OH}$ | — | — | 40 | mA |

Notes: 1. To protect chip reliability, do not exceed the output current values in table 20.3.
2. When driving a darlington pair or LED directly, always insert a current-limiting resistor in the output line, as show in figures 20.1 and 20.2.

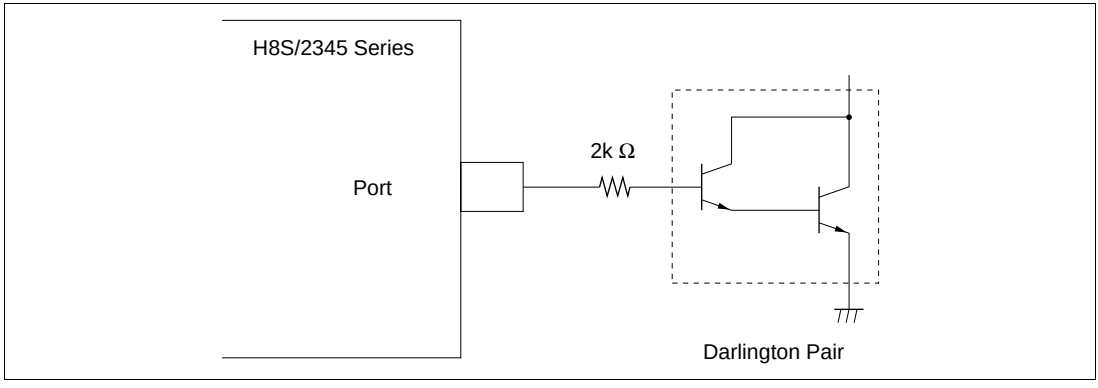



**Figure 20.1 Darlington Pair Drive Circuit (Example)**

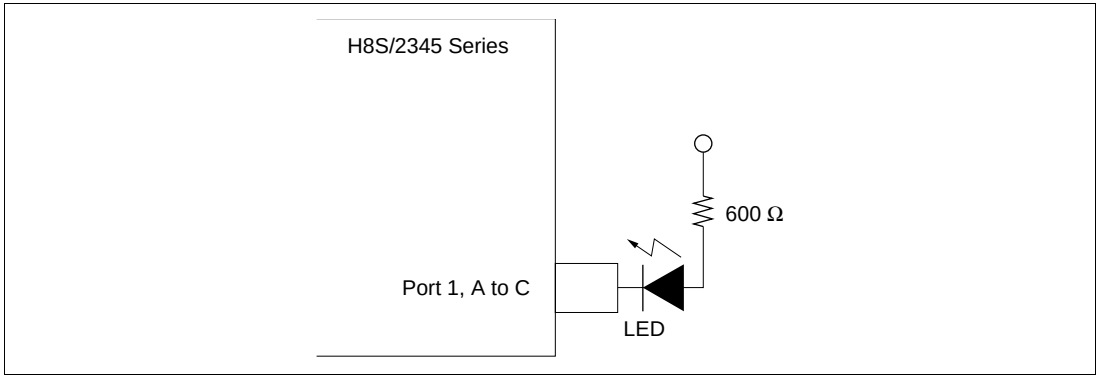



**Figure 20.2 LED Drive Circuit (Example)**

### 20.1.3 AC Characteristics

Figure 20.3 show, the test conditions for the AC characteristics.

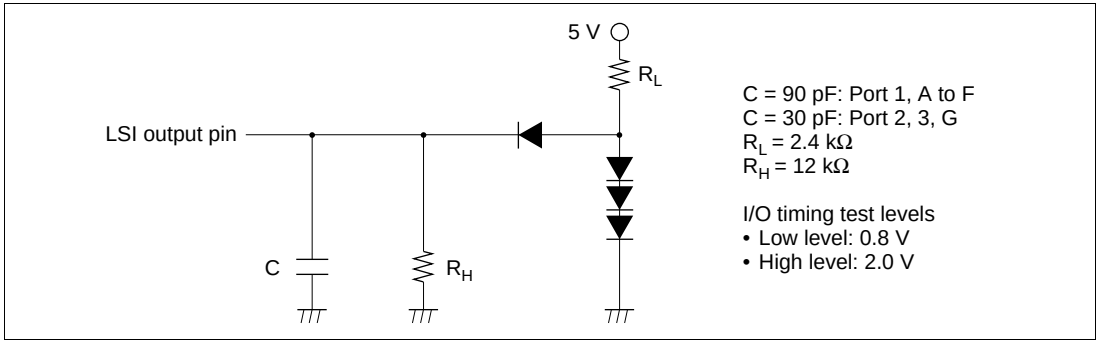



**Figure 20.3 Output Load Circuit**

**Clock Timing:** Table 20.4 lists the clock timing

**Table 20.4 Clock Timing**

Condition A: —In planning stage—
$V_{CC}$ = $AV_{CC}$ = 3.0 to 5.5 V, $V_{ref}$ = 3.0 V to $AV_{CC}$, $V_{SS}$ = $AV_{SS}$ = 0 V,
ø = 2 to 10 MHz, $T_a$ = –20 to +75°C (regular specifications),
$T_a$ = –40 to +85°C (wide-range specifications)

Condition B: $V_{CC}$ = $AV_{CC}$ = 5.0 V ± 10%, $V_{ref}$ = 4.5 V to $AV_{CC}$, $V_{SS}$ = $AV_{SS}$ = 0 V,
ø = 2 to 20 MHz, $T_a$ = –20 to +75°C (regular specifications),
$T_a$ = –40 to +85°C (wide-range specifications)

| | | Condition A | | Condition B | | | |
|---|---|---|---|---|---|---|---|
| **Item** | **Symbol** | **Min** | **Max** | **Min** | **Max** | **Unit** | **Test Conditions** |
| Clock cycle time | $t_{cyc}$ | 100 | 500 | 50 | 500 | ns | |
| Clock high pulse width | $t_{CH}$ | 35 | — | 20 | — | ns | Figure 20.7 |
| Clock low pulse width | $t_{CL}$ | 35 | — | 20 | — | ns | |
| Clock rise time | $t_{Cr}$ | — | 15 | — | 5 | ns | |
| Clock fall time | $t_{Cf}$ | — | 15 | — | 5 | ns | |
| Clock oscillator setting time at reset (crystal) | $t_{OSC1}$ | 20 | — | 10 | — | ms | Figure 20.8 |
| Clock oscillator setting time in software standby (crystal) | $t_{OSC2}$ | 8 | — | 8 | — | ms | Figure 19.2 |
| External clock output stabilization delay time | $t_{DEXT}$ | 500 | — | 500 | — | µs | Figure 20.8 |

**Control Signal Timing:** Table 20.5 lists the control signal timing.

**Table 20.5 Control Signal Timing**

Condition A: —In planning stage—
$V_{CC}$ = $AV_{CC}$ = 3.0 to 5.5 V, $V_{ref}$ = 3.0 V to $AV_{CC}$, $V_{SS}$ = $AV_{SS}$ = 0 V,
ø = 2 to 10 MHz, $T_a$ = –20 to +75°C (regular specifications),
$T_a$ = –40 to +85°C (wide-range specifications)

Condition B: $V_{CC}$ = $AV_{CC}$ = 5.0 V ± 10%, $V_{ref}$ = 4.5 V to $AV_{CC}$, $V_{SS}$ = $AV_{SS}$ = 0 V,
ø = 2 to 20 MHz, $T_a$ = –20 to +75°C (regular specifications),
$T_a$ = –40 to +85°C (wide-range specifications)

| | | Condition A | | Condition B | | | |
|---|---|---|---|---|---|---|---|
| **Item** | **Symbol** | **Min** | **Max** | **Min** | **Max** | **Unit** | **Test Conditions** |
| $\overline{RES}$ setup time | $t_{RESS}$ | 200 | — | 200 | — | ns | Figure 20.9 |
| $\overline{RES}$ pulse width | $t_{RESW}$ | 20 | — | 20 | — | $t_{cyc}$ | |
| NMI reset setup time | $t_{NMIRS}$ | 250 | — | 200 | — | ns | |
| NMI reset hold time | $t_{NMIRH}$ | 200 | — | 200 | — | ns | |
| Mode programming setup time | $t_{MDS}$ | 200 | — | 200 | — | ns | |
| NMI setup time | $t_{NMIS}$ | 250 | — | 150 | — | ns | Figure 20.10 |
| NMI hold time | $t_{NMIH}$ | 10 | — | 10 | — | ns | |
| NMI pulse width (exiting software standby mode) | $t_{NMIW}$ | 200 | — | 200 | — | ns | |
| $\overline{IRQ}$ setup time | $t_{IRQS}$ | 250 | — | 150 | — | ns | |
| $\overline{IRQ}$ hold time | $t_{IRQH}$ | 10 | — | 10 | — | ns | |
| $\overline{IRQ}$ pulse width (exiting software standby mode) | $t_{IRQW}$ | 200 | — | 200 | — | ns | |

**Bus Timing:** Table 20.6 lists the bus timing.

**Table 20.6 Bus Timing**

Condition A: —In planning stage—
$V_{CC} = AV_{CC} = 3.0$ to 5.5 V, $V_{ref} = 3.0$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 10 MHz, $T_a = -20$ to +75°C (regular specifications),
$T_a = -40$ to +85°C (wide-range specifications)

Condition B: $V_{CC} = AV_{CC} = 5.0$ V ± 10%, $V_{ref} = 4.5$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø= 2 to 20 MHz, $T_a = -20$ to +75°C (regular specifications),
$T_a = -40$ to +85°C (wide-range specifications)

| | | Condition A | | Condition B | | | |
|---|---|---|---|---|---|---|---|
| **Item** | **Symbol** | **Min** | **Max** | **Min** | **Max** | **Unit** | **Test Conditions** |
| Address delay time | $t_{AD}$ | — | 40 | — | 20 | ns | Figure 20.11 to Figure 20.15 |
| Address setup time | $t_{AS}$ | 0.5 × $t_{cyc}$ – 30 | — | 0.5 × $t_{cyc}$ – 15 | — | ns | |
| Address hold time | $t_{AH}$ | 0.5 × $t_{cyc}$ – 20 | — | 0.5 × $t_{cyc}$ – 10 | — | ns | |
| $\overline{CS}$ delay time 1 | $t_{CSD1}$ | — | 40 | — | 20 | ns | |
| $\overline{AS}$ delay time | $t_{ASD}$ | — | 40 | — | 20 | ns | |
| $\overline{RD}$ delay time 1 | $t_{RSD1}$ | — | 40 | — | 20 | ns | |
| $\overline{RD}$ delay time 2 | $t_{RSD2}$ | — | 40 | — | 20 | ns | |
| $\overline{CAS}$ delay time | $t_{CASD}$ | — | 40 | — | 20 | ns | |
| Read data setup time | $t_{RDS}$ | 30 | — | 15 | — | ns | |
| Read data hold time | $t_{RDH}$ | 0 | — | 0 | — | ns | |
| Read data access time 1 | $t_{ACC1}$ | — | 1.0 × $t_{cyc}$ – 50 | — | 1.0 × $t_{cyc}$ – 25 | ns | |
| Read data access time 2 | $t_{ACC2}$ | — | 1.5 × $t_{cyc}$ – 50 | — | 1.5 × $t_{cyc}$ – 25 | ns | |
| Read data access time 3 | $t_{ACC3}$ | — | 2.0 × $t_{cyc}$ – 50 | — | 2.0 × $t_{cyc}$ – 25 | ns | |
| Read data access time 4 | $t_{ACC4}$ | — | 2.5 × $t_{cyc}$ – 50 | — | 2.5 × $t_{cyc}$ – 25 | ns | |
| Read data access time 5 | $t_{ACC5}$ | — | 3.0 × $t_{cyc}$ – 50 | — | 3.0 × $t_{cyc}$ – 25 | ns | |

**Table 20.6 Bus Timing (cont)**

Condition A: —In planning stage—
$V_{CC} = AV_{CC} = 3.0$ to 5.5 V, $V_{ref} = 3.0$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 10 MHz, $T_a = -20$ to +75°C (regular specifications),
$T_a = -40$ to +85°C (wide-range specifications)

Condition B: $V_{CC} = AV_{CC} = 5.0$ V ± 10%, $V_{ref} = 4.5$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø= 2 to 20 MHz, $T_a = -20$ to +75°C (regular specifications),
$T_a = -40$ to +85°C (wide-range specifications)

| | | Condition A | | Condition B | | | |
|---|---|---|---|---|---|---|---|
| **Item** | **Symbol** | **Min** | **Max** | **Min** | **Max** | **Unit** | **Test Conditions** |
| $\overline{WR}$ delay time 1 | $t_{WRD1}$ | — | 40 | — | 20 | ns | Figure 20.11 to Figure 20.15 |
| $\overline{WR}$ delay time 2 | $t_{WRD2}$ | — | 40 | — | 20 | ns | |
| $\overline{WR}$ pulse width 1 | $t_{WSW1}$ | 1.0 × $t_{cyc}$ – 40 | — | 1.0 × $t_{cyc}$ – 20 | — | ns | |
| $\overline{WR}$ pulse width 2 | $t_{WSW2}$ | 1.5 × $t_{cyc}$ – 40 | — | 1.5 × $t_{cyc}$ – 20 | — | ns | |
| Write data delay time | $t_{WDD}$ | — | 60 | — | 30 | ns | |
| Write data setup time | $t_{WDS}$ | 0.5 × $t_{cyc}$ – 40 | — | 0.5 × $t_{cyc}$ – 20 | — | ns | |
| Write data hold time | $t_{WDH}$ | 0.5 × $t_{cyc}$ – 20 | — | 0.5 × $t_{cyc}$ – 10 | — | ns | |
| $\overline{WAIT}$ setup time | $t_{WTS}$ | 60 | — | 30 | — | ns | Figure 20.13 |
| $\overline{WAIT}$ hold time | $t_{WTH}$ | 10 | — | 5 | — | ns | |
| $\overline{BREQ}$ setup time | $t_{BRQS}$ | 60 | — | 30 | — | ns | Figure 20.16 |
| $\overline{BACK}$ delay time | $t_{BACD}$ | — | 30 | — | 15 | ns | |
| Bus-floating time | $t_{BZD}$ | — | 100 | — | 50 | ns | |

**Timing of On-Chip Supporting Modules:** Table 20.7 lists the timing of on-chip supporting modules.

**Table 20.7 Timing of On-Chip Supporting Modules**

Condition A: —In planning stage—
$V_{CC} = AV_{CC} = 3.0$ V to 5.5 V, $V_{ref} = 3.0$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 10 MHz, $T_a = -20$ to +75°C (regular specifications),
$T_a = -40$ to +85°C (wide-range specifications)

Condition B: $V_{CC} = AV_{CC} = 5.0$ V ± 10%, $V_{ref} = 4.5$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 20 MHz, $T_a = -20$ to +75°C (regular specifications),
$T_a = -40$ to +85°C (wide-range specifications)

| | | | | Condition A | | Condition B | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Item** | | | **Symbol** | **Min** | **Max** | **Min** | **Max** | **Unit** | **Test Conditions** |
| I/O port | Output data delay time | | $t_{PWD}$ | — | 100 | — | 50 | ns | Figure 20.17 |
| | Input data setup time | | $t_{PRS}$ | 50 | — | 30 | — | | |
| | Input data hold time | | $t_{PRH}$ | 50 | — | 30 | — | | |
| TPU | Timer output delay time | | $t_{TOCD}$ | — | 100 | — | 50 | ns | Figure 20.18 |
| | Timer input setup time | | $t_{TICS}$ | 50 | — | 30 | — | | |
| | Timer clock input setup time | | $t_{TCKS}$ | 50 | — | 30 | — | ns | Figure 20.19 |
| | Timer clock pulse width | Single edge | $t_{TCKWH}$ | 1.5 | — | 1.5 | — | $t_{cyc}$ | |
| | | Both edges | $t_{TCKWL}$ | 2.5 | — | 2.5 | — | | |
| 8-bit timer | Timer output delay time | | $t_{TMOD}$ | — | 100 | — | 50 | ns | Figure 20.20 |
| | Timer reset input setup time | | $t_{TMRS}$ | 50 | — | 30 | — | ns | Figure 20.22 |
| | Timer clock input setup time | | $t_{TMCS}$ | 50 | — | 30 | — | ns | Figure 20.21 |
| | Timer clock pulse width | Single edge | $t_{TMCWH}$ | 1.5 | — | 1.5 | — | $t_{cyc}$ | |
| | | Both edges | $t_{TMCWL}$ | 2.5 | — | 2.5 | — | | |

**Table 20.7 Timing of On-Chip Supporting Modules (cont)**

Condition A: —In planning stage—
$V_{CC} = AV_{CC} = 3.0$ V to 5.5 V, $V_{ref} = 3.0$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 10 MHz, $T_a = –20$ to +75°C (regular specifications),
$T_a = –40$ to +85°C (wide-range specifications)

Condition B: $V_{CC} = AV_{CC} = 5.0$ V ± 10%, $V_{ref} = 4.5$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 20 MHz, $T_a = –20$ to +75°C (regular specifications),
$T_a = –40$ to +85°C (wide-range specifications)

| | | | | Condition A | | Condition B | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Item** | | | **Symbol** | **Min** | **Max** | **Min** | **Max** | **Unit** | **Test Conditions** |
| SCI | Input clock cycle | Asynchronous | $t_{Scyc}$ | 4 | — | 4 | — | $t_{cyc}$ | Figure 20.24 |
| | | Synchronous | | 6 | — | 6 | — | | |
| | Input clock pulse width | | $t_{SCKW}$ | 0.4 | 0.6 | 0.4 | 0.6 | $t_{Scyc}$ | |
| | Input clock rise time | | $t_{SCKr}$ | — | 1.5 | — | 1.5 | $t_{cyc}$ | |
| | Input clock fall time | | $t_{SCKf}$ | — | 1.5 | — | 1.5 | | |
| | Transmit data delay time | | $t_{TXD}$ | — | 100 | — | 50 | ns | Figure 20.25 |
| | Receive data setup time (synchronous) | | $t_{RXS}$ | 100 | — | 50 | — | ns | |
| | Receive data hold time (synchronous) | | $t_{RXH}$ | 100 | — | 50 | — | ns | |
| A/D converter | Trigger input setup time | | $t_{TRGS}$ | 50 | — | 30 | — | ns | Figure 20.26 |

### 20.1.4 A/D Conversion Characteristics

Table 20.8 lists the A/D conversion characteristics.

**Table 20.8 A/D Conversion Characteristics**

Condition A: —In planning stage—
$V_{CC} = AV_{CC} = 3.0$ V to 5.5 V, $V_{ref} = 3.0$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 10 MHz, $T_a = –20$ to +75°C (regular specifications),
$T_a = –40$ to +85°C (wide-range specifications)

Condition B: $V_{CC} = AV_{CC} = 5.0$ V ± 10%, $V_{ref} = 4.5$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 20 MHz, $T_a = –20$ to +75°C (regular specifications),
$T_a = –40$ to +85°C (wide-range specifications)

| | Condition A | | | Condition B | | | |
|---|---|---|---|---|---|---|---|
| **Item** | **Min** | **Typ** | **Max** | **Min** | **Typ** | **Max** | **Unit** |
| Resolution | 10 | 10 | 10 | 10 | 10 | 10 | bits |
| Conversion time | — | — | 13.4 | — | — | 6.7 | µs |
| Analog input capacitance | — | — | 20 | — | — | 20 | pF |
| Permissible signal-source impedance | — | — | 5 | — | — | 10*1 | kΩ |
| | | | | — | — | 5*2 | |
| Nonlinearity error | — | — | ±7.5 | — | — | ±3.5 | LSB |
| Offset error | — | — | ±7.5 | — | — | ±3.5 | LSB |
| Full-scale error | — | — | ±7.5 | — | — | ±3.5 | LSB |
| Quantization | — | — | ±0.5 | — | — | ±0.5 | LSB |
| Absolute accuracy | — | — | ±8.0 | — | — | ±4.0 | LSB |

Notes: 1. ø ≤ 12 MHz
2. ø > 12 MHz

### 20.1.5 D/A Conversion Characteristics

Table 20.9 lists the D/A conversion characteristics.

**Table 20.9 D/A Conversion Characteristics**

Condition A: —In planning stage—
$V_{CC} = AV_{CC} = 3.0$ V to 5.5 V, $V_{ref} = 3.0$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 10 MHz, $T_a = –20$ to +75°C (regular specifications),
$T_a = –40$ to +85°C (wide-range specifications)

Condition B: $V_{CC} = AV_{CC} = 5.0$ V ± 10%, $V_{ref} = 4.5$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 20 MHz, $T_a = –20$ to +75°C (regular specifications),
$T_a = –40$ to +85°C (wide-range specifications)

| | Condition A | | | Condition B | | | | |
|---|---|---|---|---|---|---|---|---|
| **Item** | **Min** | **Typ** | **Max** | **Min** | **Typ** | **Max** | **Unit** | **Test Conditions** |
| Resolution | 8 | 8 | 8 | 8 | 8 | 8 | bit | |
| Conversion time | — | — | 10 | — | — | 10 | µs | 20-pF capacitive load |
| Absolute accuracy | — | ±2.0 | ±3.0 | — | ±1.0 | ±1.5 | LSB | 2-MΩ resistive load |
| | — | — | ±2.0 | — | — | ±1.0 | LSB | 4-MΩ resistive load |

### 20.1.6 Flash Memory Characteristics

Table 20.10 lists the flash memory characteristics.

**Table 20.10 Flash Memory Characteristics (1)**

Conditions: $V_{CC} = AV_{CC} = 4.5$ to 5.5 V, $V_{SS} = AV_{SS} = 0$ V,
$T_a = 0$ to +75°C (flash memory programming/erase operating temperature range; regular specifications) $T_a = 0$ to +85°C (flash memory programming/erase operating temperature range; wide-range specifications)

| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|
| Programming time*1, *2, *4 | | $t_P$ | — | 10 | 200 | ms/ 32 bytes | |
| Erase time*1, *3, *5 | | $t_E$ | — | 100 | 1200 | ms/block | |
| Number of programmings | | $N_{WEC}$ | — | — | 100 | Times | |
| Programming | Wait time after setting SWE bit*1 | x | 10 | — | — | µs | |
| | Wait time after setting PSU bit*1 | y | 50 | — | — | µs | |
| | Wait time after setting P bit*1, *4 | z | 150 | — | 200 | µs | |
| | Wait time after clearing P bit*1 | α | 10 | — | — | µs | |
| | Wait time after clearing PSU bit*1 | β | 10 | — | — | µs | |
| | Wait time after setting PV bit*1 | γ | 4 | — | — | µs | |
| | Wait time after H'FF dummy write*1 | ε | 2 | — | — | µs | |
| | Wait time after clearing PV bit*1 | η | 4 | — | — | µs | |
| | Max. number of programmings*1, *4 | N | — | — | 1000*5 | Times | z = 200 µs |
| Erase | Wait time after setting SWE bit*1 | x | 10 | — | — | µs | |
| | Wait time after setting ESU bit*1 | y | 200 | — | — | µs | |
| | Wait time after setting E bit*1, *6 | z | 5 | — | 10 | µs | |
| | Wait time after clearing E bit*1 | α | 10 | — | — | µs | |
| | Wait time after clearing ESU bit*1 | β | 10 | — | — | µs | |
| | Wait time after setting EV bit*1 | γ | 20 | — | — | µs | |
| | Wait time after H'FF dummy write*1 | ε | 2 | — | — | µs | |
| | Wait time after clearing EV bit*1 | η | 5 | — | — | µs | |
| | Max. number of erases*1, *5 | N | 120 | — | 240 | Times | |

Notes:
1. Time settings should be made in accordance with the programming/erase algorithm.
2. Programming time per 32 bytes. (Indicates the total time the P bit in the flash memory control register (FLMCR1) is set. The program verification time is not included.)
3. Time to erase one block. (Indicates the total time the E bit in FLMCR1 is set. The erase verification time is not included.)
4. Write time maximum value ($t_P$ (max.) = wait time after P bit setting (z) × maximum number of programmings (N)).

5. Number of times when the wait time after P bit setting (z) = 200 µs.
   The maximum number of writes (N) should be set according to the actual set value of z so as not to exceed the maximum programming time ($t_P$ (max)).
6. For the maximum erase time ($t_E$ (max)), the following relationship applies between the wait time after E bit setting (z) and the maximum number of erases (N):
   $t_E$ (max) = Wait time after E bit setting (z) × maximum number of erases (N)
   The values of z and N should be set so as to satisfy the above formula.
   Examples: When z = 5 [ms], N = 240 times
   When z = 10 [ms], N = 120 times

**Table 20.10 Flash Memory Characteristics (2) —In planning stage—**

Conditions: $V_{CC} = AV_{CC} = 3.0$ to 3.6 V, $V_{SS} = AV_{SS} = 0$ V,
$T_a = 0$ to +75°C (flash memory programming/erase operating temperature range)

| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|
| Programming time*1, *2, *4 | | $t_P$ | — | TBD | TBD | ms/ 32 bytes | |
| Erase time*1, *3, *5 | | $t_E$ | — | TBD | TBD | ms/block | |
| Number of programmings | | $N_{WEC}$ | — | — | TBD | Times | |
| Programming | Wait time after setting SWE bit*1 | x | TBD | — | — | µs | |
| | Wait time after setting PSU bit*1 | y | TBD | — | — | µs | |
| | Wait time after setting P bit*1, *4 | z | — | — | TBD | µs | |
| | Wait time after clearing P bit*1 | α | TBD | — | — | µs | |
| | Wait time after clearing PSU bit*1 | β | TBD | — | — | µs | |
| | Wait time after setting PV bit*1 | γ | TBD | — | — | µs | |
| | Wait time after H'FF dummy write*1 | ε | TBD | — | — | µs | |
| | Wait time after clearing PV bit*1 | η | TBD | — | — | µs | |
| | Max. number of programmings*1, *4 | N | — | — | TBD | Times | |
| Erase | Wait time after setting SWE bit*1 | x | TBD | — | — | µs | |
| | Wait time after setting ESU bit*1 | y | TBD | — | — | µs | |
| | Wait time after setting E bit*1, *5 | z | — | — | TBD | µs | |
| | Wait time after clearing E bit*1 | α | TBD | — | — | µs | |
| | Wait time after clearing ESU bit*1 | β | TBD | — | — | µs | |
| | Wait time after setting EV bit*1 | γ | TBD | — | | µs | |
| | Wait time after H'FF dummy write*1 | ε | TBD | — | — | µs | |
| | Wait time after clearing EV bit*1 | η | TBD | — | — | µs | |
| | Max. number of erases*1, *5 | N | — | — | TBD | Times | |

Notes: 1. Time settings should be made in accordance with the programming/erase algorithm.
2. Programming time per 32 bytes. (Indicates the total time the P bit in the flash memory control register (FLMCR1) is set. The program verification time is not included.)
3. Time to erase one block. (Indicates the total time the E bit in FLMCR1 is set. The erase

verification time is not included.)

4. Write time maximum value ($t_P$ (max.) = wait time after P bit setting (z) × maximum number of programmings (N)).
5. Erase time maximum value ($t_E$ (max.) = wait time after E bit setting (z) × maximum number of erases (N)).

## 20.2 Electrical Characteristics of ZTAT, Mask ROM, and ROMless Versions

### 20.2.1 Absolute Maximum Ratings

Table 20.11 lists the absolute maximum ratings.

**Table 20.11 Absolute Maximum Ratings**

| Item | Symbol | Value | Unit |
|---|---|---|---|
| Power supply voltage | $V_{CC}$ | –0.3 to +7.0 | V |
| Programming voltage* | $V_{PP}$ | –0.3 to +13.5 | V |
| Input voltage (except port 4) | $V_{in}$ | –0.3 to $V_{CC}$ +0.3 | V |
| Input voltage (port 4) | $V_{in}$ | –0.3 to $AV_{CC}$ +0.3 | V |
| Reference voltage | $V_{ref}$ | –0.3 to $AV_{CC}$ +0.3 | V |
| Analog power supply voltage | $AV_{CC}$ | –0.3 to +7.0 | V |
| Analog input voltage | $V_{AN}$ | –0.3 to $AV_{CC}$ +0.3 | V |
| Operating temperature | $T_{opr}$ | Regular specifications: –20 to +75 | °C |
| | | Wide-range specifications: –40 to +85 | °C |
| Storage temperature | $T_{stg}$ | –55 to +125 | °C |

Caution: Permanent damage to the chip may result if absolute maximum rating are exceeded.

Note: * ZTAT version only.

### 20.2.2 DC Characteristics

Table 20.12 lists the DC characteristics. Table 20.13 lists the permissible output currents.

**Table 20.12 DC Characteristics**

Conditions: $V_{CC} = AV_{CC} = 5.0\ V \pm 10\%$, $V_{ref} = 4.5\ V$ to $AV_{CC}$, $V_{SS} = AV_{SS} = 0\ V$*[1], $T_a = -20$ to $+75°C$ (regular specifications), $T_a = -40$ to $+85°C$ (wide-range specifications)

| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|
| Schmitt trigger input voltage | Port 2, $\overline{IRQ0}$ to $\overline{IRQ7}$ | $V_T^-$ | 1.0 | — | — | V | |
| | | $V_T^+$ | — | — | $V_{CC} \times 0.7$ | V | |
| | | $V_T^+ - V_T^-$ | 0.4 | — | — | V | |
| Input high voltage | $\overline{RES}$, $\overline{STBY}$, NMI, $MD_2$ to $MD_0$ | $V_{IH}$ | $V_{CC} - 0.7$ | — | $V_{CC} + 0.3$ | V | |
| | EXTAL | | $V_{CC} \times 0.7$ | — | $V_{CC} + 0.3$ | V | |
| | Port 1, 3, A to G | | 2.0 | — | $V_{CC} + 0.3$ | V | |
| | Port4 | | 2.0 | — | $AV_{CC} + 0.3$ | V | |
| Input low voltage | $\overline{RES}$, $\overline{STBY}$, $MD_2$ to $MD_0$ | $V_{IL}$ | –0.3 | — | 0.5 | V | |
| | NMI, EXTAL, Port 1, 3, 4, A to G | | –0.3 | — | 0.8 | V | |
| Output high voltage | All output pins | $V_{OH}$ | $V_{CC} - 0.5$ | — | — | V | $I_{OH} = -200\ \mu A$ |
| | | | 3.5 | — | — | V | $I_{OH} = -1\ mA$ |
| Output low voltage | All output pins | $V_{OL}$ | — | — | 0.4 | V | $I_{OL} = 1.6\ mA$ |
| | Port 1, A to C | | — | — | 1.0 | V | $I_{OL} = 10\ mA$ |
| Input leakage current | $\overline{RES}$ | $\lvert I_{in} \rvert$ | — | — | 10.0 | µA | $V_{in}$ = 0.5 to $V_{CC} - 0.5$ V |
| | $\overline{STBY}$, NMI, $MD_2$ to $MD_0$ | | — | — | 1.0 | µA | |
| | Port 4 | | — | — | 1.0 | µA | $V_{in}$ = 0.5 to $AV_{CC} - 0.5$ V |

Note: 1. If the A/D and D/A converters are not used, do not leave the $AV_{CC}$, $AV_{SS}$, and $V_{ref}$ pins open.
Connect $AV_{CC}$ and $V_{ref}$ to $V_{CC}$, and connect $AV_{SS}$ to $V_{SS}$.

**Table 20.12 DC Characteristics (cont)**

Conditions: $V_{CC} = AV_{CC} = 5.0\ V \pm 10\%$, $V_{ref} = 4.5\ V$ to $AV_{CC}$, $V_{SS} = AV_{SS} = 0\ V$*[1], $T_a = -20$ to $+75°C$ (regular specifications), $T_a = -40$ to $+85°C$ (wide-range specifications)

| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|
| Three-state leakage current (off state) | Port 1 to 3, A to G | $\lvert I_{TSI} \rvert$ | — | — | 1.0 | µA | $V_{in}$ = 0.5 to $V_{CC}$ – 0.5 V |
| MOS input pull-up current | Port A to E | $-I_P$ | 50 | — | 300 | µA | $V_{in} = 0\ V$ |
| Input capacitance | $\overline{RES}$ | $C_{in}$ | — | — | 80 | pF | $V_{in} = 0\ V$, f = 1 MHz, $T_a = 25°C$ |
| | NMI | | — | — | 50 | pF | |
| | All input pins except $\overline{RES}$ and NMI | | — | — | 15 | pF | |
| Current dissipation*[2] | Normal operation | $I_{CC}$*[4] | — | 60 (5.0 V) | 89 | mA | f = 20 MHz |
| | Sleep mode | | — | 40 (5.0 V) | 73 | mA | f = 20 MHz |
| | Standby mode*[3] | | — | 0.01 | 5.0 | µA | $T_a \le 50°C$ |
| | | | — | — | 20 | | $50°C < T_a$ |
| Analog power supply current | During A/D and D/A conversion | $AI_{CC}$ | — | 0.8 (5.0 V) | 2.0 | mA | |
| | Idle | | — | 0.01 | 5.0 | µA | |
| Reference current | During A/D and D/A conversion | $AI_{CC}$ | — | 1.9 (5.0 V) | 3.0 | mA | $V_{ref} = 5.0\ V$ |
| | Idle | | — | 0.01 | 5.0 | µA | |
| RAM standby voltage | | $V_{RAM}$ | 2.0 | — | — | V | |

Notes: 1. If the A/D and D/A converters are not used, do not leave the $AV_{CC}$, $AV_{SS}$, and $V_{ref}$ pins open.
Connect $AV_{CC}$ and $V_{ref}$ to $V_{CC}$, and connect $AV_{SS}$ to $V_{SS}$.

2. Current dissipation values are for $V_{IH}$ min = $V_{CC}$ –0.5 V and $V_{IL}$ max = 0.5V with all output pins unloaded and the on-chip pull-up transistors in the off state.
3. The values are for $V_{RAM} \le V_{CC} < 4.5V$, $V_{IH}$ min = $V_{CC} \times 0.9$, and $V_{IL}$ max = 0.3 V.
4. $I_{CC}$ depends on $V_{CC}$ and f as follows:
$I_{CC}$ max = 1.0 (mA) + 0.80 (mA/(MHz × V)) × $V_{CC}$ × f [normal mode]
$I_{CC}$ max = 1.0 (mA) + 0.65 (mA/(MHz × V)) × $V_{CC}$ × f [sleep mode]

**Table 20.12 DC Characteristics (cont)**

Conditions: $V_{CC}$ = $AV_{CC}$ = 2.7 to 5.5 V, $V_{ref}$ = 2.7 V to $AV_{CC}$, $V_{SS}$ = $AV_{SS}$ = 0 V*[1], $T_a$ = –20 to +75°C (regular specifications), $T_a$ = –40 to +85°C (wide-range specifications)

| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|
| Schmitt trigger input voltage | Port 2, $\overline{IRQ0}$ to $\overline{IRQ7}$ | $V_T^-$ | $V_{CC} \times 0.2$ | — | — | V | |
| | | $V_T^+$ | — | — | $V_{CC} \times 0.7$ | V | |
| | | $V_T^+ - V_T^-$ | $V_{CC} \times 0.07$ | — | — | V | |
| Input high voltage | $\overline{RES}$, $\overline{STBY}$, NMI, $MD_2$ to $MD_0$ | $V_{IH}$ | $V_{CC} \times 0.9$ | — | $V_{CC}$ +0.3 | V | |
| | EXTAL | | $V_{CC} \times 0.7$ | — | $V_{CC}$ +0.3 | V | |
| | Port 1, 3, A to G | | $V_{CC} \times 0.7$ | — | $V_{CC}$ +0.3 | V | |
| | Port 4 | | $V_{CC} \times 0.7$ | — | $AV_{CC}$ +0.3 | V | |
| Input low voltage | $\overline{RES}$, $\overline{STBY}$, $MD_2$ to $MD_0$ | $V_{IL}$ | –0.3 | — | $V_{CC} \times 0.1$ | V | |
| | NMI, EXTAL, Port 1, 3 , 4, A to G | | –0.3 | — | $V_{CC} \times 0.2$ | V | $V_{CC}$ < 4.0 V |
| | | | | | 0.8 | | $V_{CC}$ = 4.0 to 5.5 V |
| Output high voltage | All output pins | $V_{OH}$ | $V_{CC}$ – 0.5 | — | — | V | $I_{OH}$ = –200 µA |
| | | | $V_{CC}$ – 1.0 | — | — | V | $I_{OH}$ = –1 mA |
| Output low voltage | All output pins | $V_{OL}$ | — | — | 0.4 | V | $I_{OL}$ = 1.6 mA |
| | Port 1, A to C | | — | — | 1.0 | V | $V_{CC} \leq 4$ V<br>$I_{OL}$ = 5 mA<br>$4.0 < V_{CC} \leq 5.5$ V<br>$I_{OL}$ = 10 mA |
| Input leakage current | $\overline{RES}$ | $\lvert I_{in} \rvert$ | — | — | 10.0 | µA | $V_{in}$ = 0.5 to $V_{CC}$ – 0.5V |
| | $\overline{STBY}$, NMI, $MD_2$ to $MD_0$ | | — | — | 1.0 | µA | |
| | Port 4 | | — | — | 1.0 | µA | $V_{in}$ = 0.5 to $AV_{CC}$ – 0.5V |

Note: 1. If the A/D and D/A converters are not used, do not leave the $AV_{CC}$, $AV_{SS}$, and $V_{ref}$ pins open.
Connect $AV_{CC}$ and $V_{ref}$ to $V_{CC}$, and connect $AV_{SS}$ to $V_{SS}$.

**Table 20.12 DC Characteristics (cont)**

Conditions: $V_{CC}$ = $AV_{CC}$ = 2.7 to 5.5 V, $V_{ref}$ = 2.7 V to $AV_{CC}$, $V_{SS}$ = $AV_{SS}$ = 0 V*[1], $T_a$ = –20 to +75°C (regular specifications), $T_a$ = –40 to +85°C (wide-range specifications)

| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|
| Three-state leakage current (off state) | Port 1 to 3, A to G | $\lvert I_{TSI} \rvert$ | — | — | 1.0 | µA | $V_{in}$ = 0.5 to $V_{CC}$ –0.5 V |
| MOS input pull-up current | Port A to E | $-I_P$ | 10 | — | 300 | µA | $V_{CC}$ = 2.7 V to 5.5 V, $V_{in}$ = 0 V |
| Input capacitance | $\overline{RES}$ | $C_{in}$ | — | — | 80 | pF | $V_{in}$ = 0 V, f = 1 MHz, $T_a$ = 25°C |
| | NMI | | — | — | 50 | pF | |
| | All input pins except $\overline{RES}$ and NMI | | — | — | 15 | pF | |
| Current dissipation*[2] | Normal operation | $I_{CC}$*[4] | — | 18 (3.0 V) | 45 | mA | f = 10 MHz |
| | Sleep mode | | — | 11 (3.0 V) | 37 | mA | f = 10 MHz |
| | Standby mode*[3] | | — | 0.01 | 5.0 | µA | $T_a \leq 50°C$ |
| | | | — | — | 20 | | 50°C < $T_a$ |
| Analog power supply current | During A/D and D/A conversion | $AI_{CC}$ | — | 0.2 (3.0 V) | 2.0 | mA | |
| | Idle | | — | 0.01 | 5.0 | µA | |
| Reference current | During A/D and D/A conversion | $AI_{CC}$ | — | 1.2 (3.0 V) | 3.0 | mA | $V_{ref}$ = 3.0 V |
| | Idle | | — | 0.01 | 5.0 | µA | |
| RAM standby voltage | | $V_{RAM}$ | 2.0 | — | — | V | |

Notes: 1. If the A/D and D/A converters are not used, do not leave the $AV_{CC}$, $AV_{SS}$, and Vref pins open.
Connect $AV_{CC}$ and $V_{ref}$ to $V_{CC}$, and connect $AV_{SS}$ to $V_{SS}$.

2. Current dissipation values are for $V_{IH}$ min = $V_{CC}$ –0.5 V and $V_{IL}$ max = 0.5V with all output pins unloaded and the on-chip pull-up transistors in the off state.
3. The values are for $V_{RAM} \leq V_{CC} < 2.7$ V, $V_{IH}$ min = $V_{CC} \times 0.9$, and $V_{IL}$ max = 0.3V.
4. $I_{CC}$ depends on $V_{CC}$ and f as follows:
$I_{CC}$ max = 1.0 (mA) + 0.80 (mA/(MHz × V)) × $V_{CC}$ × f [normal mode]
$I_{CC}$ max = 1.0 (mA) + 0.65 (mA/(MHz × V)) × $V_{CC}$ × f [sleep mode]

**Table 20.13 Permissible Output Currents**

Conditions: $V_{CC}$ = $AV_{CC}$ = 2.7 to 5.5 V, $V_{ref}$ = 2.7 to $AV_{CC}$, $V_{SS}$ = $AV_{SS}$ = 0 V, $T_a$ = –20 to +75°C (regular specifications), $T_a$ = –40 to +85°C (wide-range specifications)

| Item | | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Permissible output low current (per pin) | Port 1, A to C | $I_{OL}$ | — | — | 10 | mA |
| | Other output pins | | — | — | 2.0 | mA |
| Permissible output low current (total) | Total of 28 pins including port 1 and A to C | $\Sigma I_{OL}$ | — | — | 80 | mA |
| | Total of all output pins, including the above | | — | — | 120 | mA |
| Permissible output high current (per pin) | All output pins | $-I_{OH}$ | — | — | 2.0 | mA |
| Permissible output high current (total) | Total of all output pins | $\Sigma -I_{OH}$ | — | — | 40 | mA |

Notes: 1. To protect chip reliability, do not exceed the output current values in table 20.13.
2. When driving a darlington pair or LED directly, always insert a current-limiting resistor in the output line, as show in figures 20.4 and 20.5.

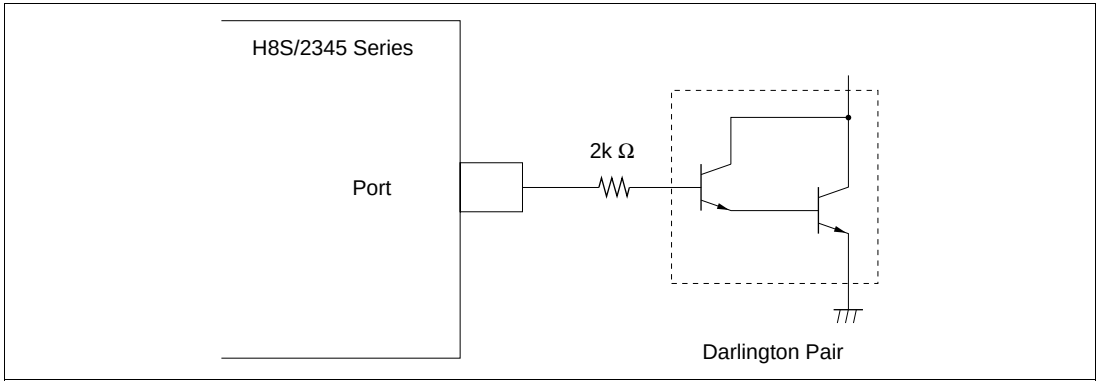



**Figure 20.4 Darlington Pair Drive Circuit (Example)**

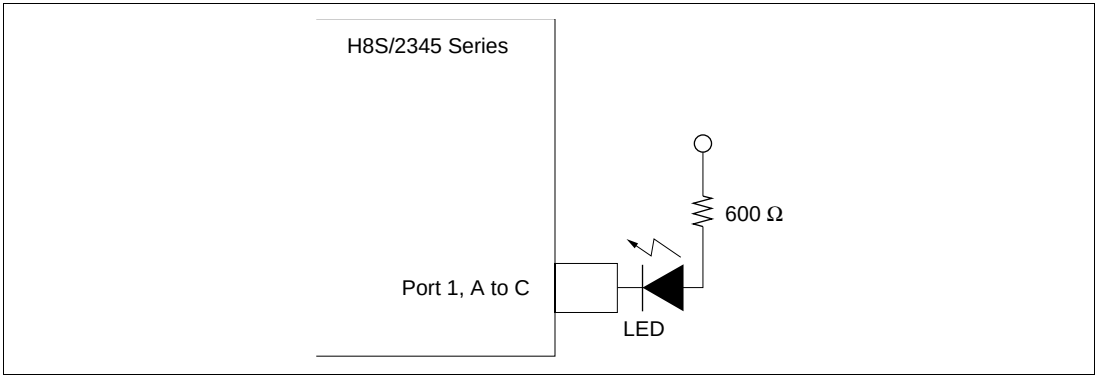



**Figure 20.5 LED Drive Circuit (Example)**

### 20.2.3 AC Characteristics

Figure 20.6 show, the test conditions for the AC characteristics.

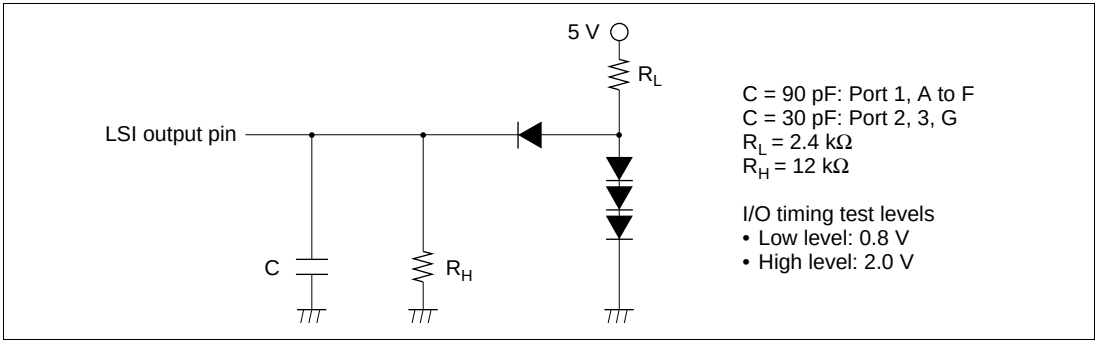



**Figure 20.6 Output Load Circuit**

**Clock Timing:** Table 20.14 lists the clock timing

**Table 20.14 Clock Timing**

Condition A: $V_{CC} = AV_{CC} = 2.7$ to 5.5 V, $V_{ref} = 2.7$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 10 MHz, $T_a = –20$ to +75°C (regular specifications),
$T_a = –40$ to +85°C (wide-range specifications)

Condition B: $V_{CC} = AV_{CC} = 5.0$ V ± 10%, $V_{ref} = 4.5$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 20 MHz, $T_a = –20$ to +75°C (regular specifications),
$T_a = –40$ to +85°C (wide-range specifications)

| | | Condition A | | Condition B | | | |
|---|---|---|---|---|---|---|---|
| **Item** | **Symbol** | **Min** | **Max** | **Min** | **Max** | **Unit** | **Test Conditions** |
| Clock cycle time | $t_{cyc}$ | 100 | 500 | 50 | 500 | ns | Figure 20.7 |
| Clock high pulse width | $t_{CH}$ | 35 | — | 20 | — | ns | |
| Clock low pulse width | $t_{CL}$ | 35 | — | 20 | — | ns | |
| Clock rise time | $t_{Cr}$ | — | 15 | — | 5 | ns | |
| Clock fall time | $t_{Cf}$ | — | 15 | — | 5 | ns | |
| Clock oscillator setting time at reset (crystal) | $t_{OSC1}$ | 20 | — | 10 | — | ms | Figure 20.8 |
| Clock oscillator setting time in software standby (crystal) | $t_{OSC2}$ | 8 | — | 8 | — | ms | Figure 19.2 |
| External clock output stabilization delay time | $t_{DEXT}$ | 500 | — | 500 | — | µs | Figure 20.8 |

**Control Signal Timing:** Table 20.15 lists the control signal timing.

**Table 20.15 Control Signal Timing**

Condition A: $V_{CC} = AV_{CC} = 2.7$ to 5.5 V, $V_{ref} = 2.7$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 10 MHz, $T_a = –20$ to +75°C (regular specifications),
$T_a = –40$ to +85°C (wide-range specifications)

Condition B: $V_{CC} = AV_{CC} = 5.0$ V ± 10%, $V_{ref} = 4.5$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 20 MHz, $T_a = –20$ to +75°C (regular specifications),
$T_a = –40$ to +85°C (wide-range specifications)

| | | Condition A | | Condition B | | | |
|---|---|---|---|---|---|---|---|
| **Item** | **Symbol** | **Min** | **Max** | **Min** | **Max** | **Unit** | **Test Conditions** |
| $\overline{RES}$ setup time | $t_{RESS}$ | 200 | — | 200 | — | ns | Figure 20.9 |
| $\overline{RES}$ pulse width | $t_{RESW}$ | 20 | — | 20 | — | $t_{cyc}$ | |
| NMI reset setup time | $t_{NMIRS}$ | 250 | — | 200 | — | ns | |
| NMI reset hold time | $t_{NMIRH}$ | 200 | — | 200 | — | ns | |
| NMI setup time | $t_{NMIS}$ | 250 | — | 150 | — | ns | Figure 20.10 |
| NMI hold time | $t_{NMIH}$ | 10 | — | 10 | — | ns | |
| NMI pulse width (exiting software standby mode) | $t_{NMIW}$ | 200 | — | 200 | — | ns | |
| $\overline{IRQ}$ setup time | $t_{IRQS}$ | 250 | — | 150 | — | ns | |
| $\overline{IRQ}$ hold time | $t_{IRQH}$ | 10 | — | 10 | — | ns | |
| $\overline{IRQ}$ pulse width (exiting software standby mode) | $t_{IRQW}$ | 200 | — | 200 | — | ns | |

**Bus Timing:** Table 20.16 lists the bus timing.

**Table 20.16 Bus Timing**

Condition A: $V_{CC} = AV_{CC} = 2.7$ to 5.5 V, $V_{ref} = 2.7$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 10 MHz, $T_a = -20$ to +75°C (regular specifications),
$T_a = -40$ to +85°C (wide-range specifications)

Condition B: $V_{CC} = AV_{CC} = 5.0$ V ± 10%, $V_{ref} = 4.5$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø= 2 to 20 MHz, $T_a = -20$ to +75°C (regular specifications),
$T_a = -40$ to +85°C (wide-range specifications)

| | | Condition A | | Condition B | | | |
|---|---|---|---|---|---|---|---|
| **Item** | **Symbol** | **Min** | **Max** | **Min** | **Max** | **Unit** | **Test Conditions** |
| Address delay time | $t_{AD}$ | — | 40 | — | 20 | ns | Figure 20.11 to Figure 20.15 |
| Address setup time | $t_{AS}$ | 0.5 × $t_{cyc}$ – 30 | — | 0.5 × $t_{cyc}$ – 15 | — | ns | |
| Address hold time | $t_{AH}$ | 0.5 × $t_{cyc}$ – 20 | — | 0.5 × $t_{cyc}$ – 10 | — | ns | |
| $\overline{CS}$ delay time 1 | $t_{CSD1}$ | — | 40 | — | 20 | ns | |
| $\overline{AS}$ delay time | $t_{ASD}$ | — | 40 | — | 20 | ns | |
| $\overline{RD}$ delay time 1 | $t_{RSD1}$ | — | 40 | — | 20 | ns | |
| $\overline{RD}$ delay time 2 | $t_{RSD2}$ | — | 40 | — | 20 | ns | |
| $\overline{CAS}$ delay time | $t_{CASD}$ | — | 40 | — | 20 | ns | |
| Read data setup time | $t_{RDS}$ | 30 | — | 15 | — | ns | |
| Read data hold time | $t_{RDH}$ | 0 | — | 0 | — | ns | |
| Read data access time 1 | $t_{ACC1}$ | — | 1.0 × $t_{cyc}$ – 50 | — | 1.0 × $t_{cyc}$ – 25 | ns | |
| Read data access time 2 | $t_{ACC2}$ | — | 1.5 × $t_{cyc}$ – 50 | — | 1.5 × $t_{cyc}$ – 25 | ns | |
| Read data access time 3 | $t_{ACC3}$ | — | 2.0 × $t_{cyc}$ – 50 | — | 2.0 × $t_{cyc}$ – 25 | ns | |
| Read data access time 4 | $t_{ACC4}$ | — | 2.5 × $t_{cyc}$ – 50 | — | 2.5 × $t_{cyc}$ – 25 | ns | |
| Read data access time 5 | $t_{ACC5}$ | — | 3.0 × $t_{cyc}$ – 50 | — | 3.0 × $t_{cyc}$ – 25 | ns | |

**Table 20.16 Bus Timing (cont)**

Condition A: $V_{CC} = AV_{CC} = 2.7$ to 5.5 V, $V_{ref} = 2.7$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V, ø = 2 to 10 MHz, $T_a = –20$ to +75°C (regular specifications), $T_a = –40$ to +85°C (wide-range specifications)

Condition B: $V_{CC} = AV_{CC} = 5.0$ V ± 10%, $V_{ref} = 4.5$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V, ø= 2 to 20 MHz, $T_a = –20$ to +75°C (regular specifications), $T_a = –40$ to +85°C (wide-range specifications)

| | | Condition A | | Condition B | | | |
|---|---|---|---|---|---|---|---|
| **Item** | **Symbol** | **Min** | **Max** | **Min** | **Max** | **Unit** | **Test Conditions** |
| $\overline{WR}$ delay time 1 | $t_{WRD1}$ | — | 40 | — | 20 | ns | Figure 20.11 to Figure 20.15 |
| $\overline{WR}$ delay time 2 | $t_{WRD2}$ | — | 40 | — | 20 | ns | |
| $\overline{WR}$ pulse width 1 | $t_{WSW1}$ | $1.0 \times t_{cyc} – 40$ | — | $1.0 \times t_{cyc} – 20$ | — | ns | |
| $\overline{WR}$ pulse width 2 | $t_{WSW2}$ | $1.5 \times t_{cyc} – 40$ | — | $1.5 \times t_{cyc} – 20$ | — | ns | |
| Write data delay time | $t_{WDD}$ | — | 60 | — | 30 | ns | |
| Write data setup time | $t_{WDS}$ | $0.5 \times t_{cyc} – 40$ | — | $0.5 \times t_{cyc} – 20$ | — | ns | |
| Write data hold time | $t_{WDH}$ | $0.5 \times t_{cyc} – 20$ | — | $0.5 \times t_{cyc} – 10$ | — | ns | |
| $\overline{WAIT}$ setup time | $t_{WTS}$ | 60 | — | 30 | — | ns | Figure 20.13 |
| $\overline{WAIT}$ hold time | $t_{WTH}$ | 10 | — | 5 | — | ns | |
| $\overline{BREQ}$ setup time | $t_{BRQS}$ | 60 | — | 30 | — | ns | Figure 20.16 |
| $\overline{BACK}$ delay time | $t_{BACD}$ | — | 30 | — | 15 | ns | |
| Bus-floating time | $t_{BZD}$ | — | 100 | — | 50 | ns | |

**Timing of On-Chip Supporting Modules:** Table 20.17 lists the timing of on-chip supporting modules.

**Table 20.17 Timing of On-Chip Supporting Modules**

Condition A: $V_{CC} = AV_{CC} = 2.7$ V to 5.5 V, $V_{ref} = 2.7$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 10 MHz, $T_a = -20$ to +75°C (regular specifications),
$T_a = -40$ to +85°C (wide-range specifications)

Condition B: $V_{CC} = AV_{CC} = 5.0$ V ± 10%, $V_{ref} = 4.5$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 20 MHz, $T_a = -20$ to +75°C (regular specifications),
$T_a = -40$ to +85°C (wide-range specifications)

| | | | | Condition A | | Condition B | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Item** | | | **Symbol** | **Min** | **Max** | **Min** | **Max** | **Unit** | **Test Conditions** |
| I/O port | Output data delay time | | $t_{PWD}$ | — | 100 | — | 50 | ns | Figure 20.17 |
| | Input data setup time | | $t_{PRS}$ | 50 | — | 30 | — | | |
| | Input data hold time | | $t_{PRH}$ | 50 | — | 30 | — | | |
| TPU | Timer output delay time | | $t_{TOCD}$ | — | 100 | — | 50 | ns | Figure 20.18 |
| | Timer input setup time | | $t_{TICS}$ | 50 | — | 30 | — | | |
| | Timer clock input setup time | | $t_{TCKS}$ | 50 | — | 30 | — | ns | Figure 20.19 |
| | Timer clock pulse width | Single edge | $t_{TCKWH}$ | 1.5 | — | 1.5 | — | $t_{cyc}$ | |
| | | Both edges | $t_{TCKWL}$ | 2.5 | — | 2.5 | — | | |
| 8-bit timer | Timer output delay time | | $t_{TMOD}$ | — | 100 | — | 50 | ns | Figure 20.20 |
| | Timer reset input setup time | | $t_{TMRS}$ | 50 | — | 30 | — | ns | Figure 20.22 |
| | Timer clock input setup time | | $t_{TMCS}$ | 50 | — | 30 | — | ns | Figure 20.21 |
| | Timer clock pulse width | Single edge | $t_{TMCWH}$ | 1.5 | — | 1.5 | — | $t_{cyc}$ | |
| | | Both edges | $t_{TMCWL}$ | 2.5 | — | 2.5 | — | | |

**Table 20.17 Timing of On-Chip Supporting Modules (cont)**

Condition A: $V_{CC} = AV_{CC} = 2.7$ V to 5.5 V, $V_{ref} = 2.7$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 10 MHz, $T_a = –20$ to +75°C (regular specifications),
$T_a = –40$ to +85°C (wide-range specifications)

Condition B: $V_{CC} = AV_{CC} = 5.0$ V ± 10%, $V_{ref} = 4.5$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 20 MHz, $T_a = –20$ to +75°C (regular specifications),
$T_a = –40$ to +85°C (wide-range specifications

| | | | | Condition A | | Condition B | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Item** | | | **Symbol** | **Min** | **Max** | **Min** | **Max** | **Unit** | **Test Conditions** |
| WDT | Overflow output delay time | | $t_{WOVD}$ | — | 100 | — | 50 | ns | Figure 20.23 |
| SCI | Input clock cycle | Asynchro-nous | $t_{Scyc}$ | 4 | — | 4 | — | $t_{cyc}$ | Figure 20.24 |
| | | Synchro-nous | | 6 | — | 6 | — | | |
| | Input clock pulse width | | $t_{SCKW}$ | 0.4 | 0.6 | 0.4 | 0.6 | $t_{Scyc}$ | |
| | Input clock rise time | | $t_{SCKr}$ | — | 1.5 | — | 1.5 | $t_{cyc}$ | |
| | Input clock fall time | | $t_{SCKf}$ | — | 1.5 | — | 1.5 | | |
| | Transmit data delay time | | $t_{TXD}$ | — | 100 | — | 50 | ns | Figure 20.25 |
| | Receive data setup time (synchronous) | | $t_{RXS}$ | 100 | — | 50 | — | ns | |
| | Receive data hold time (synchronous) | | $t_{RXH}$ | 100 | — | 50 | — | ns | |
| A/D converter | Trigger input setup time | | $t_{TRGS}$ | 50 | — | 30 | — | ns | Figure 20.26 |

### 20.2.4 A/D Conversion Characteristics

Table 20.18 lists the A/D conversion characteristics.

**Table 20.18 A/D Conversion Characteristics**

Condition A: $V_{CC} = AV_{CC} = 2.7$ to 5.5 V, $V_{ref} = 2.7$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 10 MHz, $T_a = –20$ to +75°C (regular specifications),
$T_a = –40$ to +85°C (wide-range specifications)

Condition B: $V_{CC} = AV_{CC} = 5.0$ V ± 10%, $V_{ref} = 4.5$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 20 MHz, $T_a = –20$ to +75°C (regular specifications),
$T_a = –40$ to +85°C (wide-range specifications)

| | Condition A | | | Condition B | | | |
|---|---|---|---|---|---|---|---|
| **Item** | **Min** | **Typ** | **Max** | **Min** | **Typ** | **Max** | **Unit** |
| Resolution | 10 | 10 | 10 | 10 | 10 | 10 | bits |
| Conversion time | — | — | 13.4 | — | — | 6.7 | µs |
| Analog input capacitance | — | — | 20 | — | — | 20 | pF |
| Permissible signal-source impedance | — | — | 10*[1] | — | — | 10*[3] | kΩ |
| | — | — | 5*[2] | — | — | 5*[4] | |
| Nonlinearity error | — | — | ±7.5 | — | — | ±3.5 | LSB |
| Offset error | — | — | ±7.5 | — | — | ±3.5 | LSB |
| Full-scale error | — | — | ±7.5 | — | — | ±3.5 | LSB |
| Quantization | — | — | ±0.5 | — | — | ±0.5 | LSB |
| Absolute accuracy | — | — | ±8.0 | — | — | ±4.0 | LSB |

Notes: 1. $4.0 \text{ V} \le AV_{CC} \le 5.5$ V
2. $2.7 \text{ V} \le AV_{CC} < 4.0$ V
3. ø ≤ 12 MHz
4. ø > 12 MHz

### 20.2.5 D/A Conversion Characteristics

Table 20.19 lists the D/A conversion characteristics.

**Table 20.19 D/A Conversion Characteristics**

Condition A: $V_{CC} = AV_{CC} = 2.7$ V to 5.5 V, $V_{ref} = 2.7$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 10 MHz, $T_a = –20$ to +75°C (regular specifications),
$T_a = –40$ to +85°C (wide-range specifications)

Condition B: $V_{CC} = AV_{CC} = 5.0$ V ± 10%, $V_{ref} = 4.5$ V to $AV_{CC}$, $V_{SS} = AV_{SS} = 0$ V,
ø = 2 to 20 MHz, $T_a = –20$ to +75°C (regular specifications),
$T_a = –40$ to +85°C (wide-range specifications)

| | Condition A | | | Condition B | | | | |
|---|---|---|---|---|---|---|---|---|
| **Item** | **Min** | **Typ** | **Max** | **Min** | **Typ** | **Max** | **Unit** | **Test Conditions** |
| Resolution | 8 | 8 | 8 | 8 | 8 | 8 | bit | |
| Conversion time | — | — | 10 | — | — | 10 | µs | 20-pF capacitive load |
| Absolute accuracy | — | ±2.0 | ±3.0 | — | ±1.0 | ±1.5 | LSB | 2-MΩ resistive load |
| | — | — | ±2.0 | — | — | ±1.0 | LSB | 4-MΩ resistive load |

## 20.3 Operation Timing

The operation timing is described below.

### 20.3.1 Clock Timing

The clock timing is shown below.

**System Clock Timing:** Figure 20.7 shows the system clock timing.

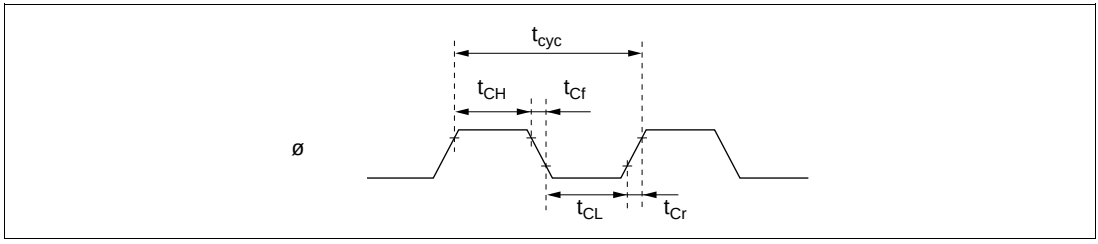



**Figure 20.7 System Clock Timing**

**Oscillator Settling Timing:** Figure 20.8 shows the oscillator settling timing.

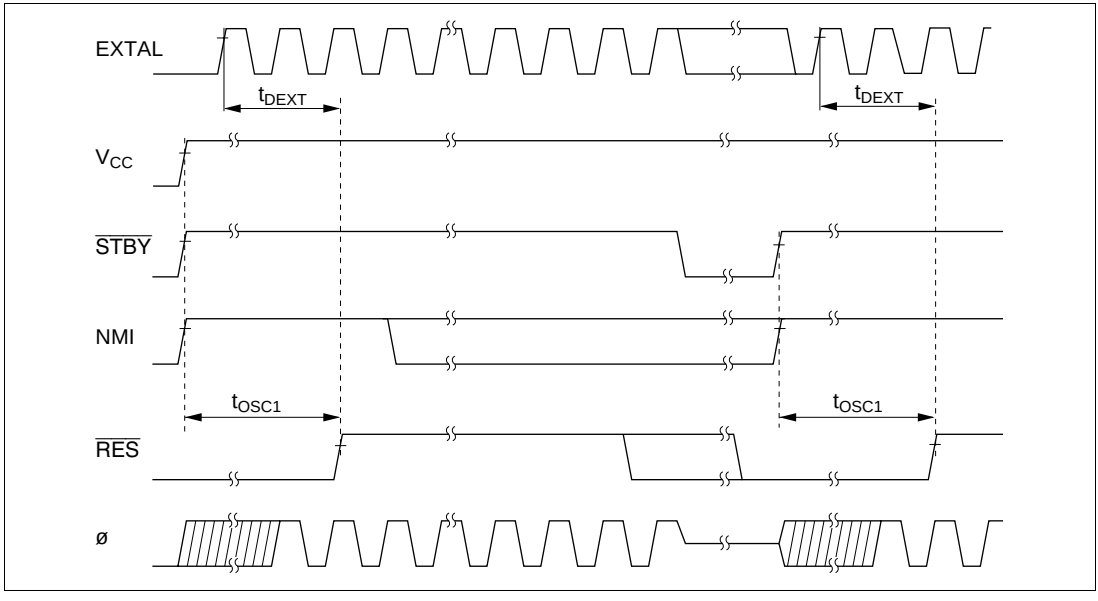



**Figure 20.8 Oscillator Settling Timing**

### 20.3.2 Control Signal Timing

The control signal timing is shown below.

**Reset Input Timing:** Figure 20.9 shows the reset input timing.

**Interrupt Input Timing:** Figure 20.10 shows the interrupt input timing for NMI and $\overline{IRQ}$.

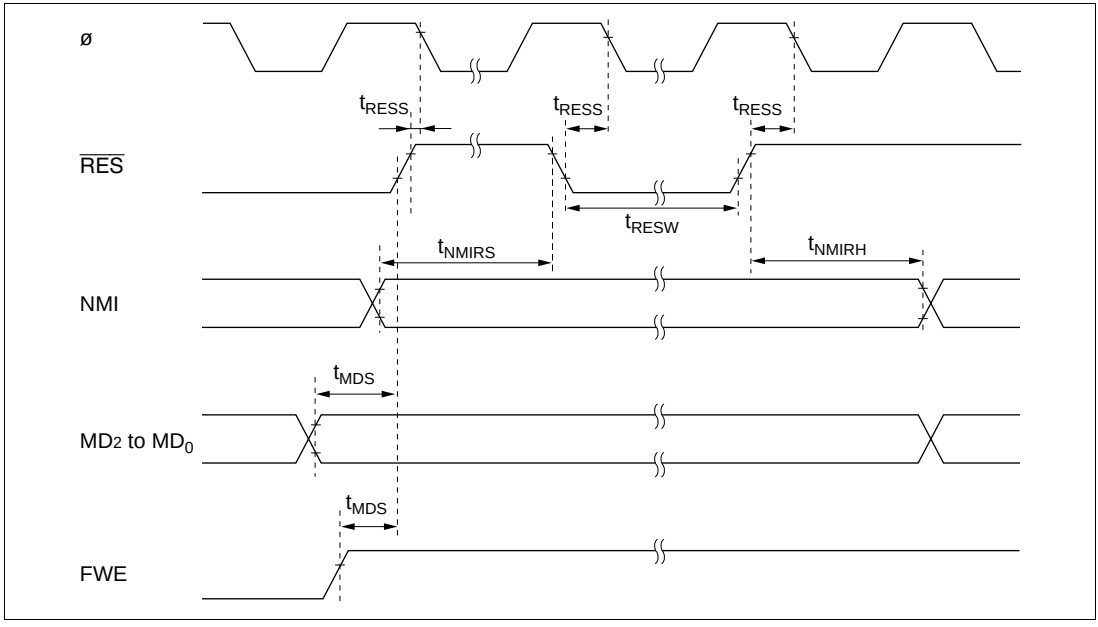



**Figure 20.9 Reset Input Timing**

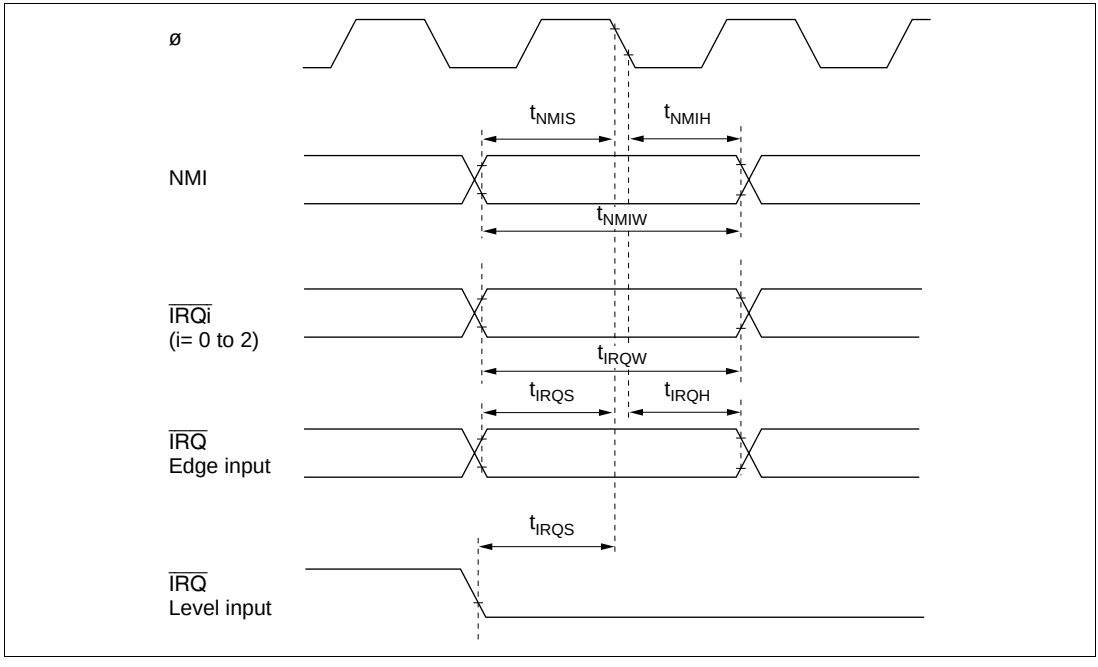



**Figure 20.10 Interrupt Input Timing**

### 20.3.3 Bus Timing

The bus timing is shown below.

**Basic Bus Timing (Two-State Access):** Figure 20.11 shows the basic bus timing for external two-state access.

**Basic Bus Timing (Three-State Access):** Figure 20.12 shows the basic bus timing for external three-state access.

**Basic Bus Timing (Three-State Access with One Wait State):** Figure 20.13 shows the basic bus timing for external three-state access with one wait state.

**Burst ROM Access Timing (Two-State Access):** Figure 20.14 shows the burst ROM access timing for two-state access.

**Burst ROM Access Timing (One-State Access):** Figure 20.15 shows the burst ROM access timing for one-state access.

**External Bus Release Timing:** Figure 20.16 shows the external bus release timing.

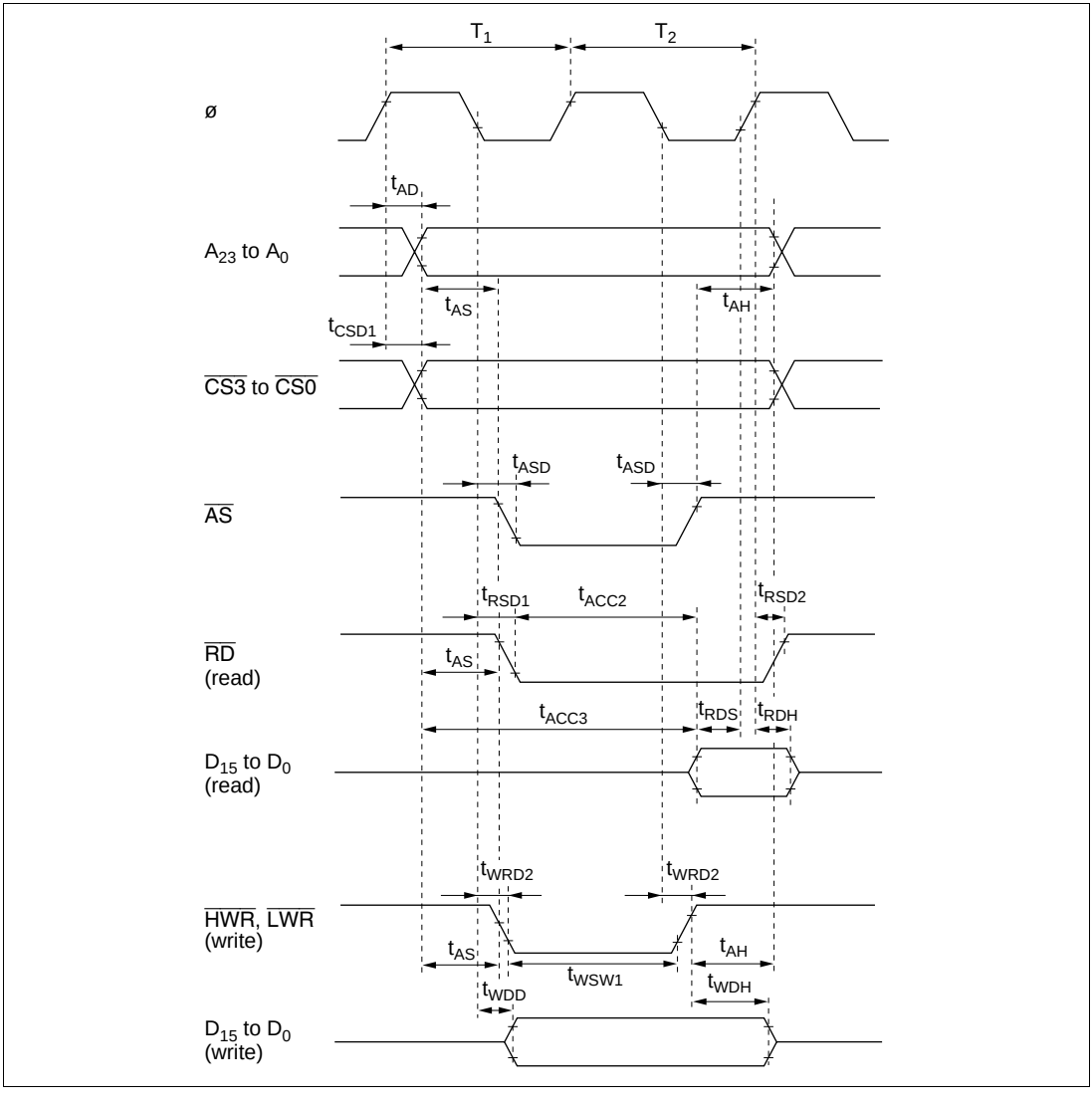



**Figure 20.11 Basic Bus Timing (Two-State Access)**

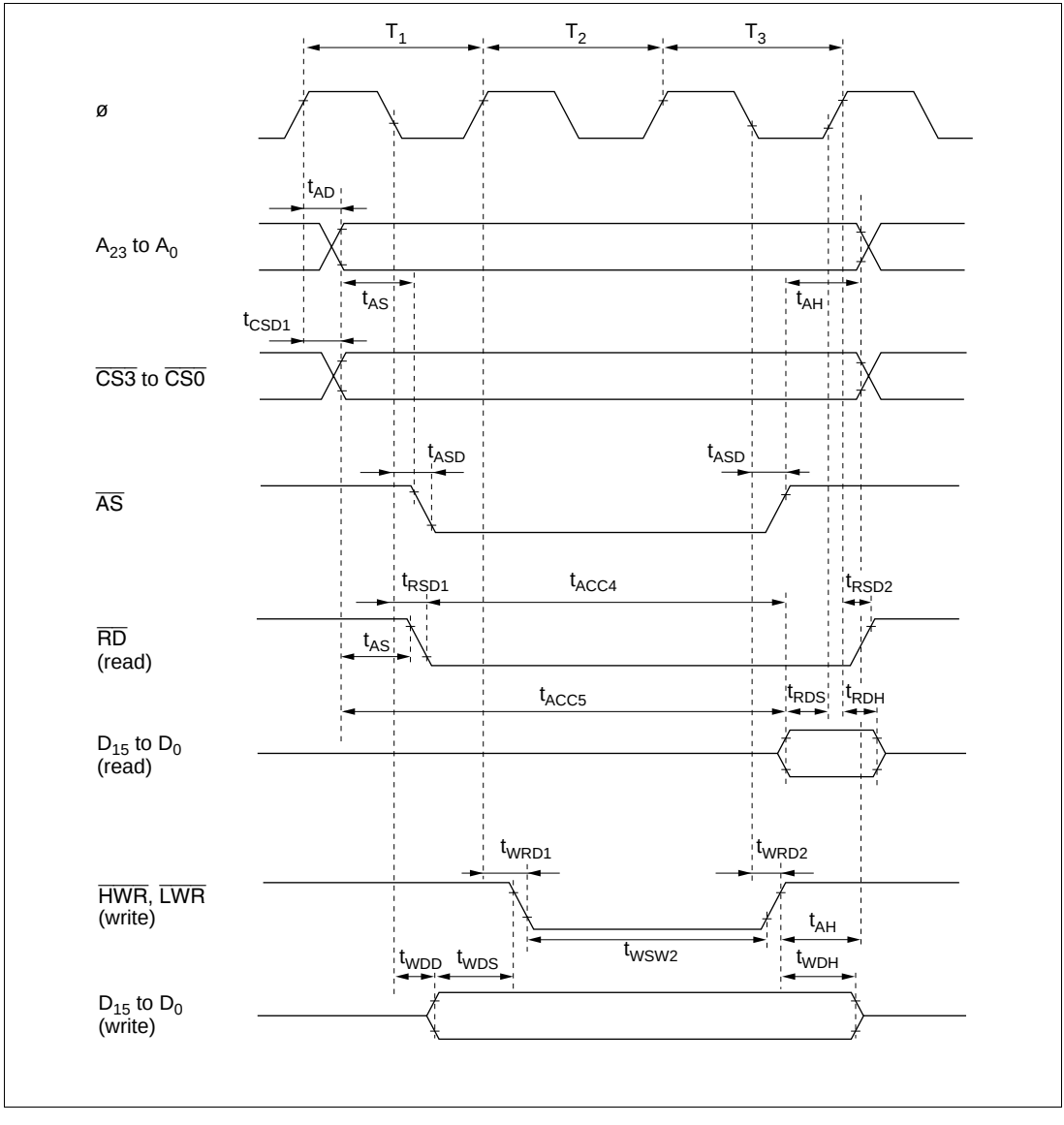



**Figure 20.12 Basic Bus Timing (Three-State Access)**

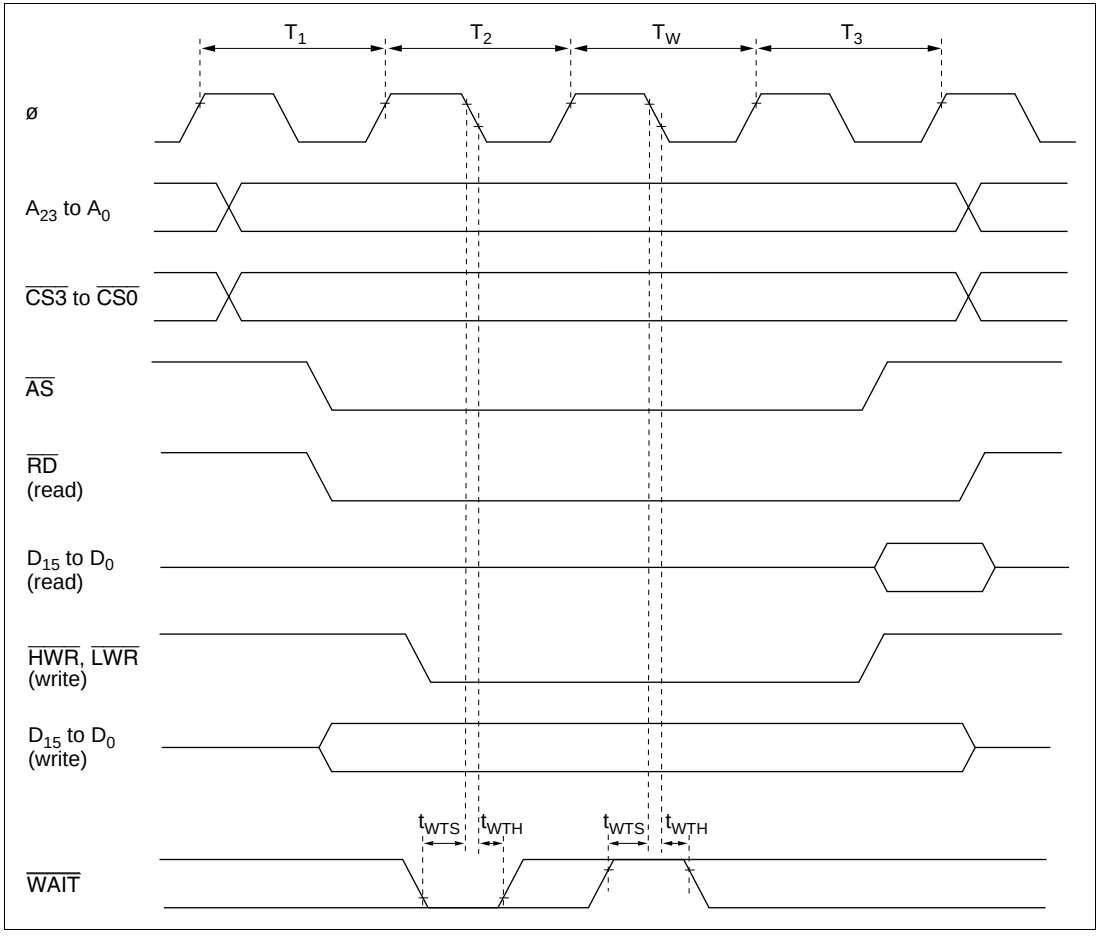



**Figure 20.13 Basic Bus Timing (Three-State Access with One Wait State)**

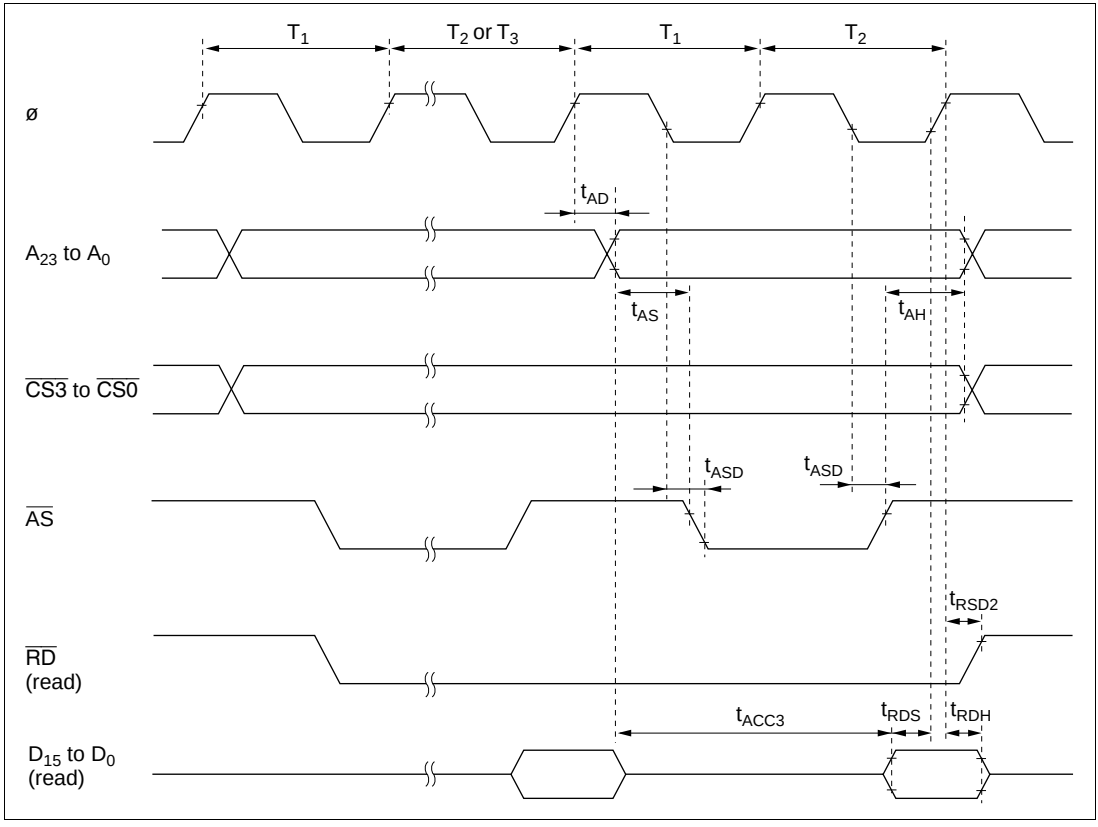



**Figure 20.14 Burst ROM Access Timing (Two-State Access)**

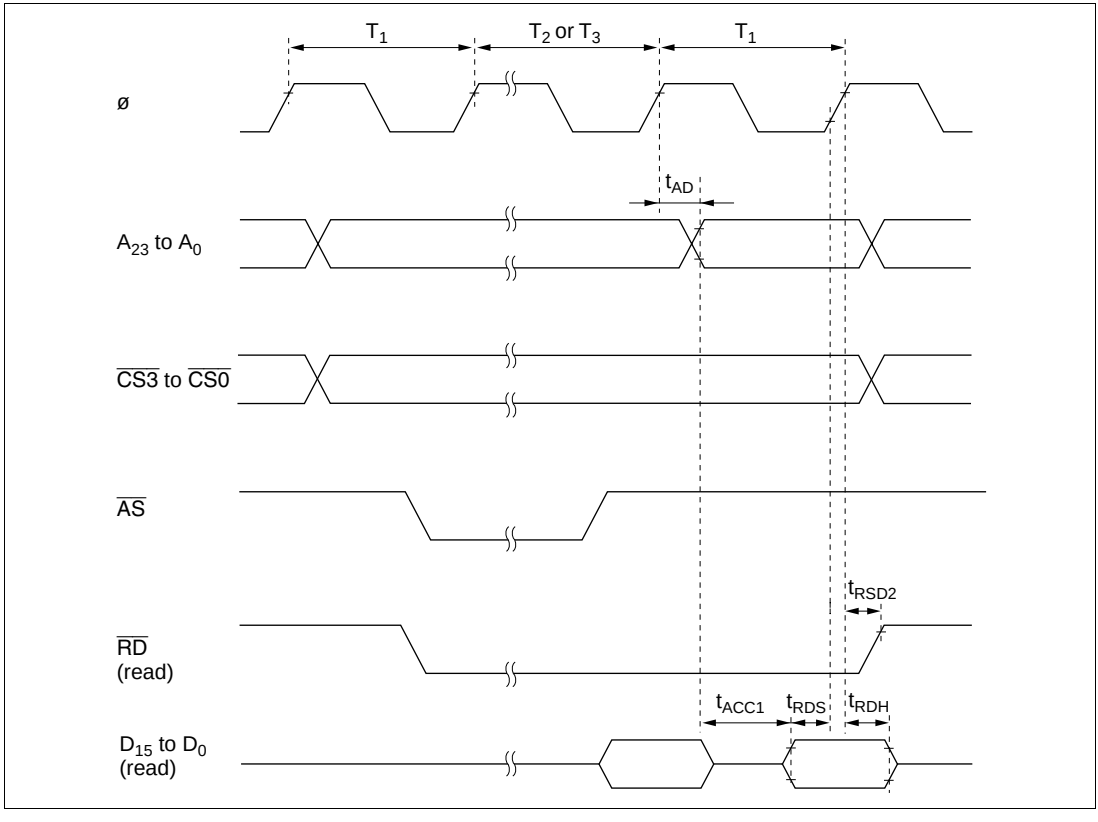



**Figure 20.15 Burst ROM Access Timing (One-State Access)**

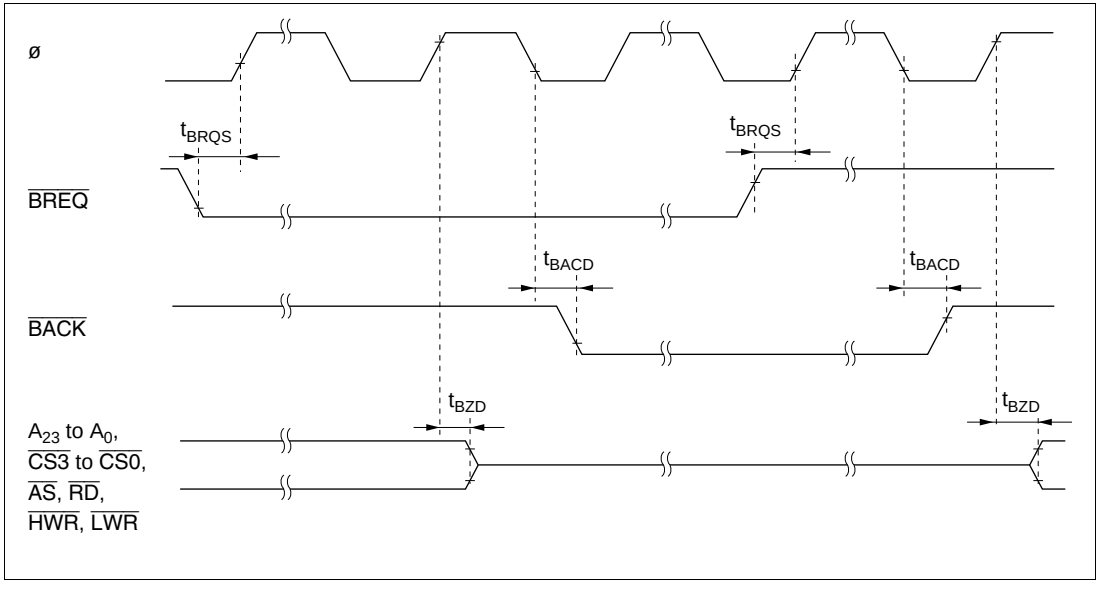



**Figure 20.16 External Bus Release Timing**

### 20.3.4 Timing for On-Chip Supporting Modules

Figure 20.17 to figure 20.26 show the timings for on-chip peripheral modules.

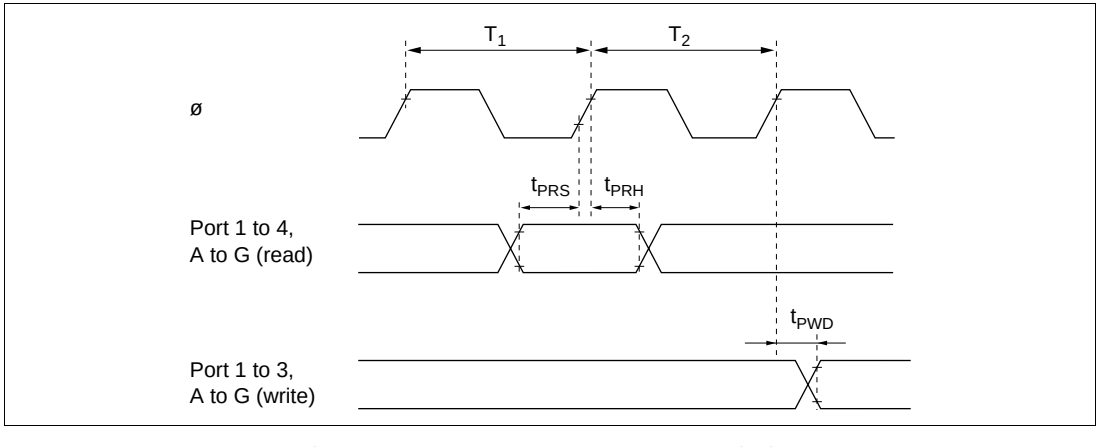



**Figure 20.17 I/O Port Input/Output Timing**

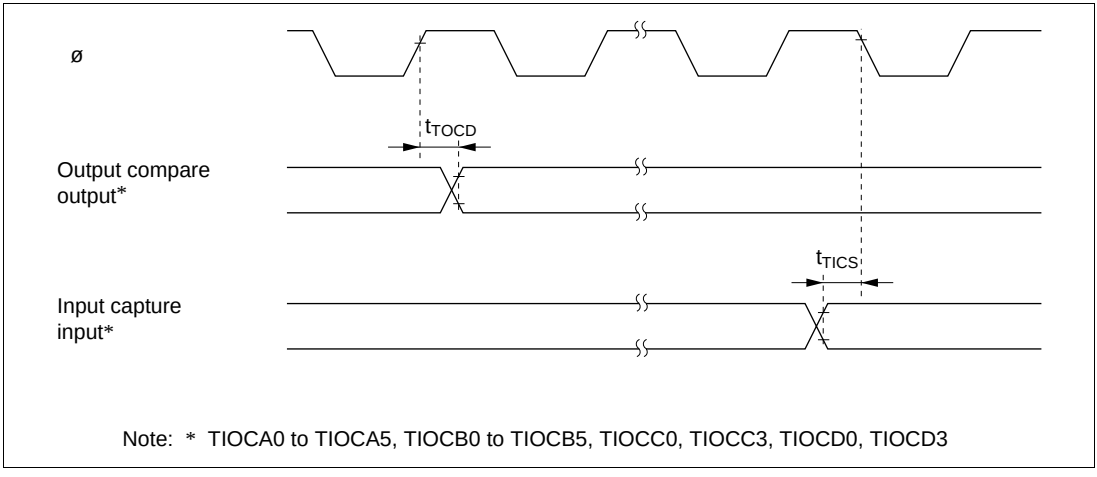



**Figure 20.18 TPU Input/Output Timing**

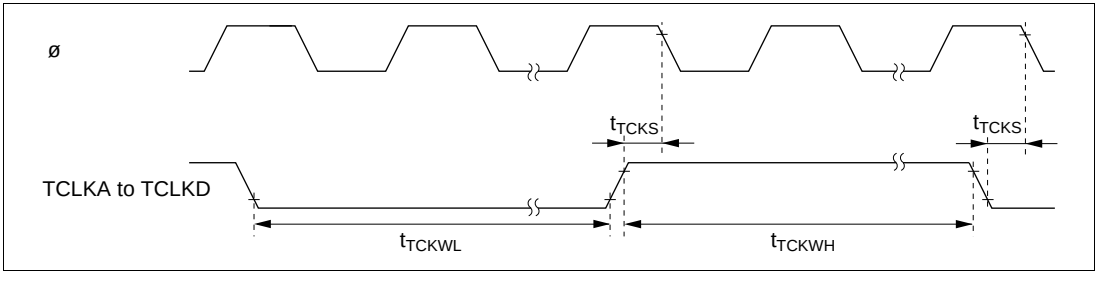



**Figure 20.19 TPU Clock Input Timing**

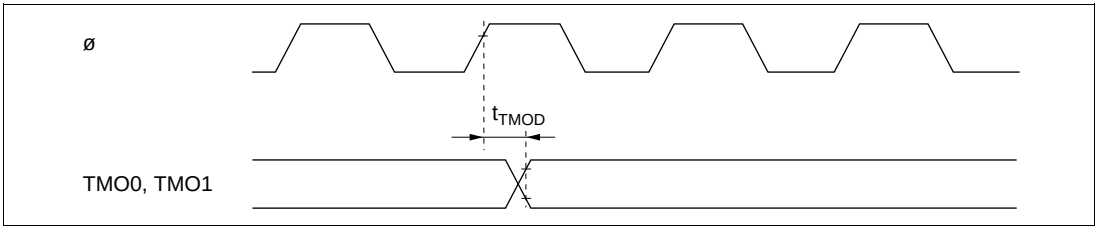



**Figure 20.20 8-Bit Timer Output Timing**

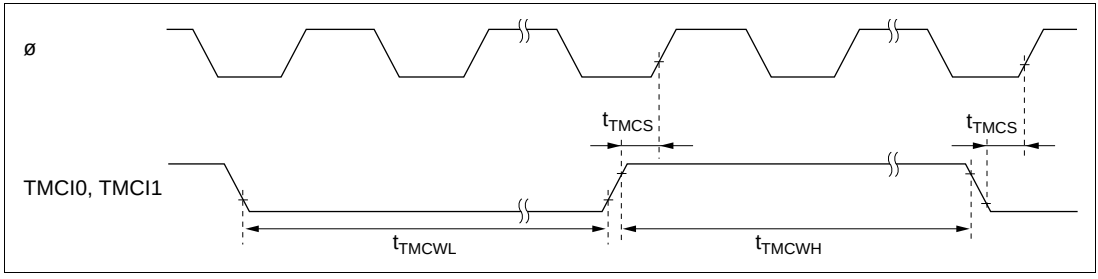



**Figure 20.21 8-Bit Timer Clock Input Timing**

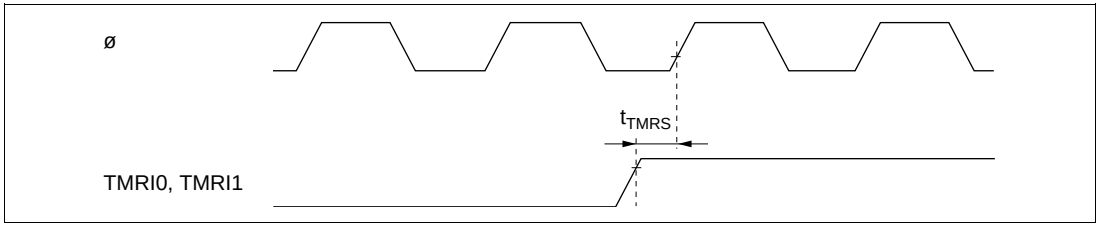



**Figure 20.22 8-Bit Timer Reset Input Timing**

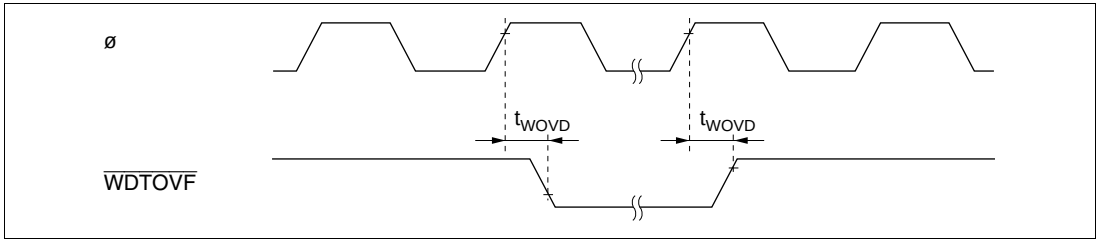



**Figure 20.23 WDT Output Timing**
**(ZTAT version, Mask ROM version, and ROMless version only)**

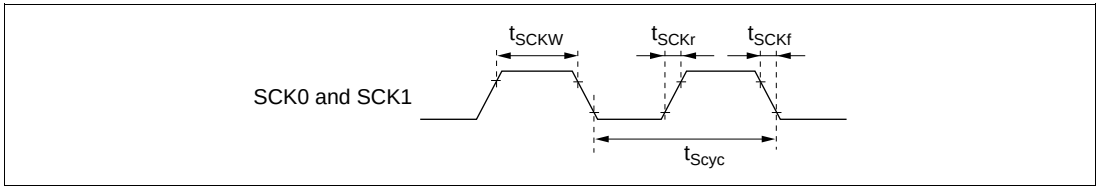



**Figure 20.24 SCK Clock Input Timing**

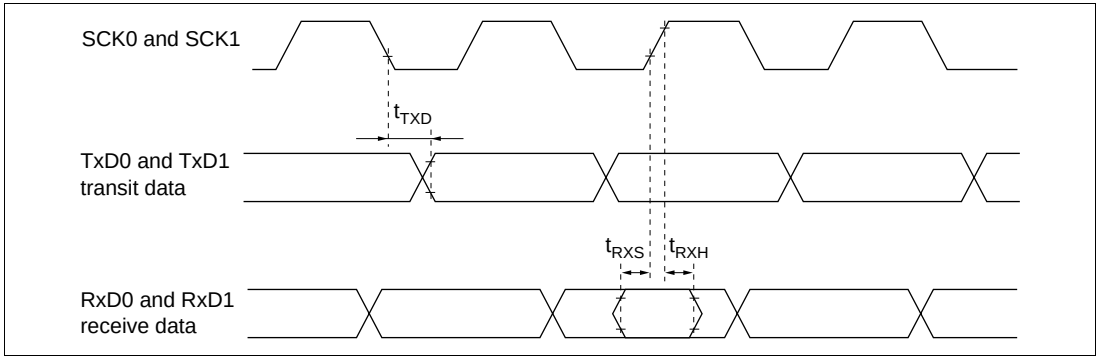



**Figure 20.25 SCI Input/Output Timing (Clock Synchronous Mode)**

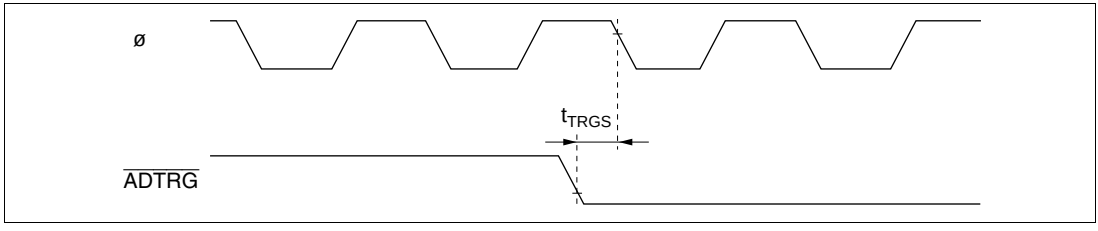



**Figure 20.26 A/D Converter External Trigger Input Timing**

## 20.4 Usage Note

Although the F-ZTAT, ZTAT, mask ROM, and ROMless versions fully meet the electrical specifications listed in this manual, due to differences in the fabrication process, the on-chip ROM, and the layout patterns, there will be differences in the actual values of the electrical characteristics, the operating margins, the noise margins, and other aspects.

Therefore, if a system is estimated using the F-ZTAT or ZTAT version, a similar evaluation should also be performed using the mask ROM version.

# Appendix A Instruction Set

## A.1 Instruction List

### Operand Notation

| | |
|---|---|
| Rd | General register (destination)*[1] |
| Rs | General register (source)*[1] |
| Rn | General register*[1] |
| ERn | General register (32-bit register) |
| MAC | Multiply-and-accumulate register (32-bit register)*[2] |
| (EAd) | Destination operand |
| (EAs) | Source operand |
| EXR | Extended control register |
| CCR | Condition-code register |
| N | N (negative) flag in CCR |
| Z | Z (zero) flag in CCR |
| V | V (overflow) flag in CCR |
| C | C (carry) flag in CCR |
| PC | Program counter |
| SP | Stack pointer |
| #IMM | Immediate data |
| disp | Displacement |
| + | Add |
| – | Subtract |
| × | Multiply |
| ÷ | Divide |
| ∧ | Logical AND |
| ∨ | Logical OR |
| ⊕ | Logical exclusive OR |
| → | Transfer from the operand on the left to the operand on the right, or transition from the state on the left to the state on the right |
| ¬ | Logical NOT (logical complement) |
| ( ) < > | Contents of operand |
| :8/:16/:24/:32 | 8-, 16-, 24-, or 32-bit length |

Notes: 1. General registers include 8-bit registers (R0H to R7H, R0L to R7L), 16-bit registers (R0 to R7, E0 to E7), and 32-bit registers (ER0 to ER7).

2. The MAC register cannot be used in the H8S/2345 Series.

## Condition Code Notation

| Symbol | |
|---|---|
| ↕ | Changes according to the result of instruction |
| * | Undetermined (no guaranteed value) |
| 0 | Always cleared to 0 |
| 1 | Always set to 1 |
| — | Not affected by execution of the instruction |

## Table A.1 Instruction Set

### (1) Data Transfer Instructions

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | | Operation | Condition Code | | | | | | No. of States*1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | \| | | I | H | N | Z | V | C | Advanced |
| MOV | MOV.B #xx:8,Rd | B | 2 | | | | | | | | | #xx:8→Rd8 | — | — | ↕ | ↕ | 0 | — | 1 |
| | MOV.B Rs,Rd | B | | 2 | | | | | | | | Rs8→Rd8 | — | — | ↕ | ↕ | 0 | — | 1 |
| | MOV.B @ERs,Rd | B | | | 2 | | | | | | | @ERs→Rd8 | — | — | ↕ | ↕ | 0 | — | 2 |
| | MOV.B @(d:16,ERs),Rd | B | | | | 4 | | | | | | @(d:16,ERs)→Rd8 | — | — | ↕ | ↕ | 0 | — | 3 |
| | MOV.B @(d:32,ERs),Rd | B | | | | 8 | | | | | | @(d:32,ERs)→Rd8 | — | — | ↕ | ↕ | 0 | — | 5 |
| | MOV.B @ERs+,Rd | B | | | | | 2 | | | | | @ERs→Rd8,ERs32+1→ERs32 | — | — | ↕ | ↕ | 0 | — | 3 |
| | MOV.B @aa:8,Rd | B | | | | | | 2 | | | | @aa:8→Rd8 | — | — | ↕ | ↕ | 0 | — | 2 |
| | MOV.B @aa:16,Rd | B | | | | | | 4 | | | | @aa:16→Rd8 | — | — | ↕ | ↕ | 0 | — | 3 |
| | MOV.B @aa:32,Rd | B | | | | | | 6 | | | | @aa:32→Rd8 | — | — | ↕ | ↕ | 0 | — | 4 |
| | MOV.B Rs,@ERd | B | | | 2 | | | | | | | Rs8→@ERd | — | — | ↕ | ↕ | 0 | — | 2 |
| | MOV.B Rs,@(d:16,ERd) | B | | | | 4 | | | | | | Rs8→@(d:16,ERd) | — | — | ↕ | ↕ | 0 | — | 3 |
| | MOV.B Rs,@(d:32,ERd) | B | | | | 8 | | | | | | Rs8→@(d:32,ERd) | — | — | ↕ | ↕ | 0 | — | 5 |
| | MOV.B Rs,@-ERd | B | | | | | 2 | | | | | ERd32-1→ERd32,Rs8→@ERd | — | — | ↕ | ↕ | 0 | — | 3 |
| | MOV.B Rs,@aa:8 | B | | | | | | 2 | | | | Rs8→@aa:8 | — | — | ↕ | ↕ | 0 | — | 2 |
| | MOV.B Rs,@aa:16 | B | | | | | | 4 | | | | Rs8→@aa:16 | — | — | ↕ | ↕ | 0 | — | 3 |
| | MOV.B Rs,@aa:32 | B | | | | | | 6 | | | | Rs8→@aa:32 | — | — | ↕ | ↕ | 0 | — | 4 |
| | MOV.W #xx:16,Rd | W | 4 | | | | | | | | | #xx:16→Rd16 | — | — | ↕ | ↕ | 0 | — | 2 |
| | MOV.W Rs,Rd | W | | 2 | | | | | | | | Rs16→Rd16 | — | — | ↕ | ↕ | 0 | — | 1 |
| | MOV.W @ERs,Rd | W | | | 2 | | | | | | | @ERs→Rd16 | — | — | ↕ | ↕ | 0 | — | 2 |

**Table A.1 Instruction Set (cont)**

**(1) Data Transfer Instructions (cont)**

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | \| | Operation | Condition Code I | H | N | Z | V | C | No. of States*1 Advanced |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MOV | MOV.W @(d:16,ERs),Rd | W | | | | 4 | | | | | | @(d:16,ERs)→Rd16 | — | — | ↕ | ↕ | 0 | — | 3 |
| | MOV.W @(d:32,ERs),Rd | W | | | | 8 | | | | | | @(d:32,ERs)→Rd16 | — | — | ↕ | ↕ | 0 | — | 5 |
| | MOV.W @ERs+,Rd | W | | | | | 2 | | | | | @ERs→Rd16,ERs32+2→ERs32 | — | — | ↕ | ↕ | 0 | — | 3 |
| | MOV.W @aa:16,Rd | W | | | | | | 4 | | | | @aa:16→Rd16 | — | — | ↕ | ↕ | 0 | — | 3 |
| | MOV.W @aa:32,Rd | W | | | | | | 6 | | | | @aa:32→Rd16 | — | — | ↕ | ↕ | 0 | — | 4 |
| | MOV.W Rs,@ERd | W | | | 2 | | | | | | | Rs16→@ERd | — | — | ↕ | ↕ | 0 | — | 2 |
| | MOV.W Rs,@(d:16,ERd) | W | | | | 4 | | | | | | Rs16→@(d:16,ERd) | — | — | ↕ | ↕ | 0 | — | 3 |
| | MOV.W Rs,@(d:32,ERd) | W | | | | 8 | | | | | | Rs16→@(d:32,ERd) | — | — | ↕ | ↕ | 0 | — | 5 |
| | MOV.W Rs,@-ERd | W | | | | | 2 | | | | | ERd32-2→ERd32,Rs16→@ERd | — | — | ↕ | ↕ | 0 | — | 3 |
| | MOV.W Rs,@aa:16 | W | | | | | | 4 | | | | Rs16→@aa:16 | — | — | ↕ | ↕ | 0 | — | 3 |
| | MOV.W Rs,@aa:32 | W | | | | | | 6 | | | | Rs16→@aa:32 | — | — | ↕ | ↕ | 0 | — | 4 |
| | MOV.L #xx:32,ERd | L | 6 | | | | | | | | | #xx:32→ERd32 | — | — | ↕ | ↕ | 0 | — | 3 |
| | MOV.L ERs,ERd | L | | 2 | | | | | | | | ERs32→ERd32 | — | — | ↕ | ↕ | 0 | — | 1 |
| | MOV.L @ERs,ERd | L | | | 4 | | | | | | | @ERs→ERd32 | — | — | ↕ | ↕ | 0 | — | 4 |
| | MOV.L @(d:16,ERs),ERd | L | | | | 6 | | | | | | @(d:16,ERs)→ERd32 | — | — | ↕ | ↕ | 0 | — | 5 |
| | MOV.L @(d:32,ERs),ERd | L | | | | 10 | | | | | | @(d:32,ERs)→ERd32 | — | — | ↕ | ↕ | 0 | — | 7 |
| | MOV.L @ERs+,ERd | L | | | | | 4 | | | | | @ERs→ERd32,ERs32+4→@ERs32 | — | — | ↕ | ↕ | 0 | — | 5 |
| | MOV.L @aa:16,ERd | L | | | | | | 6 | | | | @aa:16→ERd32 | — | — | ↕ | ↕ | 0 | — | 5 |
| | MOV.L @aa:32,ERd | L | | | | | | 8 | | | | @aa:32→ERd32 | — | — | ↕ | ↕ | 0 | — | 6 |

**Table A.1 Instruction Set (cont)**

**(1) Data Transfer Instructions (cont)**

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | | Operation | Condition Code | | | | | | No. of States*1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | \| | | I | H | N | Z | V | C | Advanced |
| MOV | MOV.L ERs,@ERd | L | | | 4 | | | | | | | ERs32→@ERd | — | — | ↕ | ↕ | 0 | — | 4 |
| | MOV.L ERs,@(d:16,ERd) | L | | | | 6 | | | | | | ERs32→@(d:16,ERd) | — | — | ↕ | ↕ | 0 | — | 5 |
| | MOV.L ERs,@(d:32,ERd) | L | | | | 10 | | | | | | ERs32→@(d:32,ERd) | — | — | ↕ | ↕ | 0 | — | 7 |
| | MOV.L ERs,@-ERd | L | | | | | 4 | | | | | ERd32-4→ERd32,ERs32→@ERd | — | — | ↕ | ↕ | 0 | — | 5 |
| | MOV.L ERs,@aa:16 | L | | | | | | 6 | | | | ERs32→@aa:16 | — | — | ↕ | ↕ | 0 | — | 5 |
| | MOV.L ERs,@aa:32 | L | | | | | | 8 | | | | ERs32→@aa:32 | — | — | ↕ | ↕ | 0 | — | 6 |
| POP | POP.W Rn | W | | | | | | | | | 2 | @SP→Rn16,SP+2→SP | — | — | ↕ | ↕ | 0 | — | 3 |
| | POP.L ERn | L | | | | | | | | | 4 | @SP→ERn32,SP+4→SP | — | — | ↕ | ↕ | 0 | — | 5 |
| PUSH | PUSH.W Rn | W | | | | | | | | | 2 | SP-2→SP,Rn16→@SP | — | — | ↕ | ↕ | 0 | — | 3 |
| | PUSH.L ERn | L | | | | | | | | | 4 | SP-4→SP,ERn32→@SP | — | — | ↕ | ↕ | 0 | — | 5 |
| LDM | LDM @SP+,(ERm-ERn) | L | | | | | | | | | 4 | (@SP→ERn32,SP+4→SP)<br>Repeated for each register restored | — | — | — | — | — | — | 7/9/11 [1] |
| STM | STM (ERm-ERn),@-SP | L | | | | | | | | | 4 | (SP-4→SP,ERn32→@SP)<br>Repeated for each register saved | — | — | — | — | — | — | 7/9/11 [1] |
| MOVFPE | MOVFPE @aa:16,Rd | Cannot be used in the H8S/2345 Series | | | | | | | | | | | | | | | | | [2] |
| MOVTPE | MOVTPE Rs,@aa:16 | Cannot be used in the H8S/2345 Series | | | | | | | | | | | | | | | | | [2] |

## Table A.1 Instruction Set

### (2) Arithmetic Instructions

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | \| | Operation | Condition Code I | H | N | Z | V | C | No. of States*1 Advanced |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ADD | ADD.B #xx:8,Rd | B | 2 | | | | | | | | | Rd8+#xx:8→Rd8 | — | ↕ | ↕ | ↕ | ↕ | ↕ | 1 |
| | ADD.B Rs,Rd | B | | 2 | | | | | | | | Rd8+Rs8→Rd8 | — | ↕ | ↕ | ↕ | ↕ | ↕ | 1 |
| | ADD.W #xx:16,Rd | W | 4 | | | | | | | | | Rd16+#xx:16→Rd16 | — | [3] | ↕ | ↕ | ↕ | ↕ | 2 |
| | ADD.W Rs,Rd | W | | 2 | | | | | | | | Rd16+Rs16→Rd16 | — | [3] | ↕ | ↕ | ↕ | ↕ | 1 |
| | ADD.L #xx:32,ERd | L | 6 | | | | | | | | | ERd32+#xx:32→ERd32 | — | [4] | ↕ | ↕ | ↕ | ↕ | 3 |
| | ADD.L ERs,ERd | L | | 2 | | | | | | | | ERd32+ERs32→ERd32 | — | [4] | ↕ | ↕ | ↕ | ↕ | 1 |
| ADDX | ADDX #xx:8,Rd | B | 2 | | | | | | | | | Rd8+#xx:8+C→Rd8 | — | ↕ | ↕ | [5] | ↕ | ↕ | 1 |
| | ADDX Rs,Rd | B | | 2 | | | | | | | | Rd8+Rs8+C→Rd8 | — | ↕ | ↕ | [5] | ↕ | ↕ | 1 |
| ADDS | ADDS #1,ERd | L | | 2 | | | | | | | | ERd32+1→ERd32 | — | — | — | — | — | — | 1 |
| | ADDS #2,ERd | L | | 2 | | | | | | | | ERd32+2→ERd32 | — | — | — | — | — | — | 1 |
| | ADDS #4,ERd | L | | 2 | | | | | | | | ERd32+4→ERd32 | — | — | — | — | — | — | 1 |
| INC | INC.B Rd | B | | 2 | | | | | | | | Rd8+1→Rd8 | — | — | ↕ | ↕ | ↕ | — | 1 |
| | INC.W #1,Rd | W | | 2 | | | | | | | | Rd16+1→Rd16 | — | — | ↕ | ↕ | ↕ | — | 1 |
| | INC.W #2,Rd | W | | 2 | | | | | | | | Rd16+2→Rd16 | — | — | ↕ | ↕ | ↕ | — | 1 |
| | INC.L #1,ERd | L | | 2 | | | | | | | | ERd32+1→ERd32 | — | — | ↕ | ↕ | ↕ | — | 1 |
| | INC.L #2,ERd | L | | 2 | | | | | | | | ERd32+2→ERd32 | — | — | ↕ | ↕ | ↕ | — | 1 |
| DAA | DAA Rd | B | | 2 | | | | | | | | Rd8 decimal adjust→Rd8 | — | * | ↕ | ↕ | * | ↕ | 1 |
| SUB | SUB.B Rs,Rd | B | | 2 | | | | | | | | Rd8-Rs8→Rd8 | — | ↕ | ↕ | ↕ | ↕ | ↕ | 1 |
| | SUB.W #xx:16,Rd | W | 4 | | | | | | | | | Rd16-#xx:16→Rd16 | — | [3] | ↕ | ↕ | ↕ | ↕ | 2 |

**Table A.1 Instruction Set (cont)**

**(2) Arithmetic Instructions (cont)**

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | | Operation | Condition Code | | | | | | No. of States*1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | \| | | I | H | N | Z | V | C | Advanced |
| SUB | SUB.W Rs,Rd | W | | 2 | | | | | | | | Rd16-Rs16→Rd16 | — | [3] | ↕ | ↕ | ↕ | ↕ | 1 |
| | SUB.L #xx:32,ERd | L | 6 | | | | | | | | | ERd32-#xx:32→ERd32 | — | [4] | ↕ | ↕ | ↕ | ↕ | 3 |
| | SUB.L ERs,ERd | L | | 2 | | | | | | | | ERd32-ERs32→ERd32 | — | [4] | ↕ | ↕ | ↕ | ↕ | 1 |
| SUBX | SUBX #xx:8,Rd | B | 2 | | | | | | | | | Rd8-#xx:8-C→Rd8 | — | ↕ | ↕ | [5] | ↕ | ↕ | 1 |
| | SUBX Rs,Rd | B | | 2 | | | | | | | | Rd8-Rs8-C→Rd8 | — | ↕ | ↕ | [5] | ↕ | ↕ | 1 |
| SUBS | SUBS #1,ERd | L | | 2 | | | | | | | | ERd32-1→ERd32 | — | — | — | — | — | — | 1 |
| | SUBS #2,ERd | L | | 2 | | | | | | | | ERd32-2→ERd32 | — | — | — | — | — | — | 1 |
| | SUBS #4,ERd | L | | 2 | | | | | | | | ERd32-4→ERd32 | — | — | — | — | — | — | 1 |
| DEC | DEC.B Rd | B | | 2 | | | | | | | | Rd8-1→Rd8 | — | — | ↕ | ↕ | ↕ | — | 1 |
| | DEC.W #1,Rd | W | | 2 | | | | | | | | Rd16-1→Rd16 | — | — | ↕ | ↕ | ↕ | — | 1 |
| | DEC.W #2,Rd | W | | 2 | | | | | | | | Rd16-2→Rd16 | — | — | ↕ | ↕ | ↕ | — | 1 |
| | DEC.L #1,ERd | L | | 2 | | | | | | | | ERd32-1→ERd32 | — | — | ↕ | ↕ | ↕ | — | 1 |
| | DEC.L #2,ERd | L | | 2 | | | | | | | | ERd32-2→ERd32 | — | — | ↕ | ↕ | ↕ | — | 1 |
| DAS | DAS Rd | B | | 2 | | | | | | | | Rd8 decimal adjust→Rd8 | — | * | ↕ | ↕ | * | — | 1 |
| MULXU | MULXU.B Rs,Rd | B | | 2 | | | | | | | | Rd8×Rs8→Rd16 (unsigned multiplication) | — | — | — | — | — | — | 12 |
| | MULXU.W Rs,ERd | W | | 2 | | | | | | | | Rd16×Rs16→ERd32 (unsigned multiplication) | — | — | — | — | — | — | 20 |
| MULXS | MULXS.B Rs,Rd | B | | 4 | | | | | | | | Rd8×Rs8→Rd16 (signed multiplication) | — | — | ↕ | ↕ | — | — | 13 |
| | MULXS.W Rs,ERd | W | | 4 | | | | | | | | Rd16×Rs16→ERd32 (signed multiplication) | — | — | ↕ | ↕ | — | — | 21 |

**Table A.1 Instruction Set (cont)**

**(2) Arithmetic Instructions (cont)**

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | \| | Operation | Condition Code I | H | N | Z | V | C | No. of States*1 Advanced |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DIVXU | DIVXU.B Rs,Rd | B | | 2 | | | | | | | | Rd16÷Rs8→Rd16 (RdH: remainder, RdL: quotient) (unsigned division) | — | — | [6] | [7] | — | — | 12 |
| | DIVXU.W Rs,ERd | W | | 2 | | | | | | | | ERd32÷Rs16→ERd32 (Ed: remainder, Rd: quotient) (unsigned division) | — | — | [6] | [7] | — | — | 20 |
| DIVXS | divxs.B Rs,Rd | B | | 4 | | | | | | | | Rd16÷Rs8→Rd16 (RdH: remainder, RdL: quotient) (signed division) | — | — | [8] | [7] | — | — | 13 |
| | DIVXS.W Rs,ERd | W | | 4 | | | | | | | | ERd32÷Rs16→ERd32 (Ed: remainder, Rd: quotient) (signed division) | — | — | [8] | [7] | — | — | 21 |
| CMP | CMP.B #xx:8,Rd | B | 2 | | | | | | | | | Rd8-#xx:8 | — | ↕ | ↕ | ↕ | ↕ | ↕ | 1 |
| | CMP.B Rs,Rd | B | | 2 | | | | | | | | Rd8-Rs8 | — | ↕ | ↕ | ↕ | ↕ | ↕ | 1 |
| | CMP.W #xx:16,Rd | W | 4 | | | | | | | | | Rd16-#xx:16 | — | [3] | ↕ | ↕ | ↕ | ↕ | 2 |
| | CMP.W Rs,Rd | W | | 2 | | | | | | | | Rd16-Rs16 | — | [3] | ↕ | ↕ | ↕ | ↕ | 1 |
| | CMP.L #xx:32,ERd | L | 6 | | | | | | | | | ERd32-#xx:32 | — | [4] | ↕ | ↕ | ↕ | ↕ | 3 |
| | CMP.L ERs,ERd | L | | 2 | | | | | | | | ERd32-ERs32 | — | [4] | ↕ | ↕ | ↕ | ↕ | 1 |
| NEG | NEG.B Rd | B | | 2 | | | | | | | | 0-Rd8→Rd8 | — | ↕ | ↕ | ↕ | ↕ | ↕ | 1 |
| | NEG.W Rd | W | | 2 | | | | | | | | 0-Rd16→Rd16 | — | ↕ | ↕ | ↕ | ↕ | ↕ | 1 |
| | NEG.L ERd | L | | 2 | | | | | | | | 0-ERd32→ERd32 | — | ↕ | ↕ | ↕ | ↕ | ↕ | 1 |
| EXTU | EXTU.W Rd | W | | 2 | | | | | | | | 0→(<bit 15 to 8> of Rd16) | — | — | 0 | ↕ | 0 | — | 1 |
| | EXTU.L ERd | L | | 2 | | | | | | | | 0→(<bit 31 to 16> of ERd32) | — | — | 0 | ↕ | 0 | — | 1 |

**Table A.1 Instruction Set (cont)**

**(2) Arithmetic Instructions (cont)**

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | | Operation | Condition Code | | | | | | No. of States*1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | \| | | I | H | N | Z | V | C | Advanced |
| EXTS | EXTS.W Rd | W | | 2 | | | | | | | | (<bit 7> of Rd16)→ (<bit 15 to 8> of Rd16) | — | — | ↕ | ↕ | 0 | — | 1 |
| | EXTS.L ERd | L | | 2 | | | | | | | | (<bit 15> of ERd32)→ (<bit 31 to 16> of ERd32) | — | — | ↕ | ↕ | 0 | — | 1 |
| TAS | TAS @ERd | B | | | 4 | | | | | | | @ERd-0→CCR set, (1)→ (<bit 7> of @ERd) | — | — | ↕ | ↕ | 0 | — | 4 |
| MAC | MAC @ERn+, @ERm+ | Cannot be used in the H8S/2345 Series | | | | | | | | | | | | | | | | | [2] |
| CLRMAC | CLRMAC | | | | | | | | | | | | | | | | | | |
| LDMAC | LDMAC ERs,MACH | | | | | | | | | | | | | | | | | | |
| | LDMAC ERs,MACL | | | | | | | | | | | | | | | | | | |
| STMAC | STMAC MACH,ERd | | | | | | | | | | | | | | | | | | |
| | STMAC MACL,ERd | | | | | | | | | | | | | | | | | | |

**Table A.1 Instruction Set**

**(3) Logical Instructions**

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | | Operation | Condition Code | | | | | | No. of States*1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | \| | | I | H | N | Z | V | C | Advanced |
| AND | AND.B #xx:8,Rd | B | 2 | | | | | | | | | Rd8∧#xx:8→Rd8 | — | — | ↕ | ↕ | 0 | — | 1 |
| | AND.B Rs,Rd | B | | 2 | | | | | | | | Rd8∧Rs8→Rd8 | — | — | ↕ | ↕ | 0 | — | 1 |
| | AND.W #xx:16,Rd | W | 4 | | | | | | | | | Rd16∧#xx:16→Rd16 | — | — | ↕ | ↕ | 0 | — | 2 |
| | AND.W Rs,Rd | W | | 2 | | | | | | | | Rd16∧Rs16→Rd16 | — | — | ↕ | ↕ | 0 | — | 1 |
| | AND.L #xx:32,ERd | L | 6 | | | | | | | | | ERd32∧#xx:32→ERd32 | — | — | ↕ | ↕ | 0 | — | 3 |
| | AND.L ERs,ERd | L | | 4 | | | | | | | | ERd32∧ERs32→ERd32 | — | — | ↕ | ↕ | 0 | — | 2 |
| OR | OR.B #xx:8,Rd | B | 2 | | | | | | | | | Rd8∨#xx:8→Rd8 | — | — | ↕ | ↕ | 0 | — | 1 |
| | OR.B Rs,Rd | B | | 2 | | | | | | | | Rd8∨Rs8→Rd8 | — | — | ↕ | ↕ | 0 | — | 1 |
| | OR.W #xx:16,Rd | W | 4 | | | | | | | | | Rd16∨#xx:16→Rd16 | — | — | ↕ | ↕ | 0 | — | 2 |
| | OR.W Rs,Rd | W | | 2 | | | | | | | | Rd16∨Rs16→Rd16 | — | — | ↕ | ↕ | 0 | — | 1 |
| | OR.L #xx:32,ERd | L | 6 | | | | | | | | | ERd32∨#xx:32→ERd32 | — | — | ↕ | ↕ | 0 | — | 3 |
| | OR.L ERs,ERd | L | | 4 | | | | | | | | ERd32∨ERs32→ERd32 | — | — | ↕ | ↕ | 0 | — | 2 |
| XOR | XOR.B #xx:8,Rd | B | 2 | | | | | | | | | Rd8⊕#xx:8→Rd8 | — | — | ↕ | ↕ | 0 | — | 1 |
| | XOR.B Rs,Rd | B | | 2 | | | | | | | | Rd8⊕Rs8→Rd8 | — | — | ↕ | ↕ | 0 | — | 1 |
| | XOR.W #xx:16,Rd | W | 4 | | | | | | | | | Rd16⊕#xx:16→Rd16 | — | — | ↕ | ↕ | 0 | — | 2 |
| | XOR.W Rs,Rd | W | | 2 | | | | | | | | Rd16⊕Rs16→Rd16 | — | — | ↕ | ↕ | 0 | — | 1 |
| | XOR.L #xx:32,ERd | L | 6 | | | | | | | | | ERd32⊕#xx:32→ERd32 | — | — | ↕ | ↕ | 0 | — | 3 |
| | XOR.L ERs,ERd | L | | 4 | | | | | | | | ERd32⊕ERs32→ERd32 | — | — | ↕ | ↕ | 0 | — | 2 |
| NOT | NOT.B Rd | B | | 2 | | | | | | | | ¬ Rd8→Rd8 | — | — | ↕ | ↕ | 0 | — | 1 |
| | NOT.W Rd | W | | 2 | | | | | | | | ¬ Rd16→Rd16 | — | — | ↕ | ↕ | 0 | — | 1 |
| | NOT.L ERd | L | | 2 | | | | | | | | ¬ ERd32→ERd32 | — | — | ↕ | ↕ | 0 | — | 1 |

## Table A.1 Instruction Set

### (4) Shift Instructions

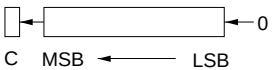
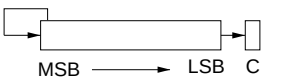
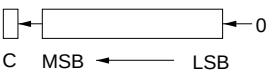
| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | — | Operation | Condition Code I | H | N | Z | V | C | No. of States*1 Advanced |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SHAL | SHAL.B Rd | B | | 2 | | | | | | | |   | — | — | ↕ | ↕ | ↕ | ↕ | 1 |
| | SHAL.B #2,Rd | B | | 2 | | | | | | | | | — | — | ↕ | ↕ | ↕ | ↕ | 1 |
| | SHAL.W Rd | W | | 2 | | | | | | | | | — | — | ↕ | ↕ | ↕ | ↕ | 1 |
| | SHAL.W #2,Rd | W | | 2 | | | | | | | | | — | — | ↕ | ↕ | ↕ | ↕ | 1 |
| | SHAL.L ERd | L | | 2 | | | | | | | | | — | — | ↕ | ↕ | ↕ | ↕ | 1 |
| | SHAL.L #2,ERd | L | | 2 | | | | | | | | | — | — | ↕ | ↕ | ↕ | ↕ | 1 |
| SHAR | SHAR.B Rd | B | | 2 | | | | | | | |   | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | SHAR.B #2,Rd | B | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | SHAR.W Rd | W | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | SHAR.W #2,Rd | W | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | SHAR.L ERd | L | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | SHAR.L #2,ERd | L | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| SHLL | SHLL.B Rd | B | | 2 | | | | | | | |   | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | SHLL.B #2,Rd | B | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | SHLL.W Rd | W | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | SHLL.W #2,Rd | W | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | SHLL.L ERd | L | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | SHLL.L #2,ERd | L | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |

**Table A.1 Instruction Set (cont)**

**(4) Shift Instructions (cont)**

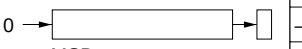
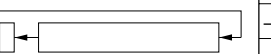
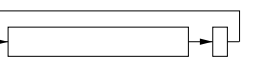
| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | — | Operation | Condition Code I | H | N | Z | V | C | No. of States*1 Advanced |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SHLR | SHLR.B Rd | B | | 2 | | | | | | | |  | — | — | 0 | ↕ | 0 | ↕ | 1 |
| | SHLR.B #2,Rd | B | | 2 | | | | | | | | | — | — | 0 | ↕ | 0 | ↕ | 1 |
| | SHLR.W Rd | W | | 2 | | | | | | | | | — | — | 0 | ↕ | 0 | ↕ | 1 |
| | SHLR.W #2,Rd | W | | 2 | | | | | | | | | — | — | 0 | ↕ | 0 | ↕ | 1 |
| | SHLR.L ERd | L | | 2 | | | | | | | | | — | — | 0 | ↕ | 0 | ↕ | 1 |
| | SHLR.L #2,ERd | L | | 2 | | | | | | | | | — | — | 0 | ↕ | 0 | ↕ | 1 |
| ROTXL | ROTXL.B Rd | B | | 2 | | | | | | | |  | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | ROTXL.B #2,Rd | B | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | ROTXL.W Rd | W | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | ROTXL.W #2,Rd | W | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | ROTXL.L ERd | L | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | ROTXL.L #2,ERd | L | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| ROTXR | ROTXR.B Rd | B | | 2 | | | | | | | |  | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | ROTXR.B #2,Rd | B | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | ROTXR.W Rd | W | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | ROTXR.W #2,Rd | W | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | ROTXR.L ERd | L | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | ROTXR.L #2,ERd | L | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |

**Table A.1 Instruction Set (cont)**

**(4) Shift Instructions (cont)**

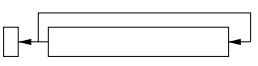
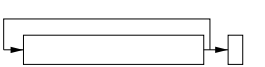
| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | | Operation | Condition Code | | | | | | No. of States*1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | \| | | I | H | N | Z | V | C | Advanced |
| ROTL | ROTL.B Rd | B | | 2 | | | | | | | |   | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | ROTL.B #2,Rd | B | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | ROTL.W Rd | W | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | ROTL.W #2,Rd | W | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | ROTL.L ERd | L | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | ROTL.L #2,ERd | L | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| ROTR | ROTR.B Rd | B | | 2 | | | | | | | |   | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | ROTR.B #2,Rd | B | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | ROTR.W Rd | W | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | ROTR.W #2,Rd | W | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | ROTR.L ERd | L | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |
| | ROTR.L #2,ERd | L | | 2 | | | | | | | | | — | — | ↕ | ↕ | 0 | ↕ | 1 |

**Table A.1 Instruction Set**

**(5) Bit-Manipulation Instructions**

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | | Operation | Condition Code | | | | | | No. of States*1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | \| | | I | H | N | Z | V | C | Advanced |
| BSET | BSET #xx:3,Rd | B | | 2 | | | | | | | | (#xx:3 of Rd8)←1 | — | — | — | — | — | — | 1 |
| | BSET #xx:3,@ERd | B | | | 4 | | | | | | | (#xx:3 of @ERd)←1 | — | — | — | — | — | — | 4 |
| | BSET #xx:3,@aa:8 | B | | | | | | 4 | | | | (#xx:3 of @aa:8)←1 | — | — | — | — | — | — | 4 |
| | BSET #xx:3,@aa:16 | B | | | | | | 6 | | | | (#xx:3 of @aa:16)←1 | — | — | — | — | — | — | 5 |
| | BSET #xx:3,@aa:32 | B | | | | | | 8 | | | | (#xx:3 of @aa:32)←1 | — | — | — | — | — | — | 6 |
| | BSET Rn,Rd | B | | 2 | | | | | | | | (Rn8 of Rd8)←1 | — | — | — | — | — | — | 1 |
| | BSET Rn,@ERd | B | | | 4 | | | | | | | (Rn8 of @ERd)←1 | — | — | — | — | — | — | 4 |
| | BSET Rn,@aa:8 | B | | | | | | 4 | | | | (Rn8 of @aa:8)←1 | — | — | — | — | — | — | 4 |
| | BSET Rn,@aa:16 | B | | | | | | 6 | | | | (Rn8 of @aa:16)←1 | — | — | — | — | — | — | 5 |
| | BSET Rn,@aa:32 | B | | | | | | 8 | | | | (Rn8 of @aa:32)←1 | — | — | — | — | — | — | 6 |
| BCLR | BCLR #xx:3,Rd | B | | 2 | | | | | | | | (#xx:3 of Rd8)←0 | — | — | — | — | — | — | 1 |
| | BCLR #xx:3,@ERd | B | | | 4 | | | | | | | (#xx:3 of @ERd)←0 | — | — | — | — | — | — | 4 |
| | BCLR #xx:3,@aa:8 | B | | | | | | 4 | | | | (#xx:3 of @aa:8)←0 | — | — | — | — | — | — | 4 |
| | BCLR #xx:3,@aa:16 | B | | | | | | 6 | | | | (#xx:3 of @aa:16)←0 | — | — | — | — | — | — | 5 |
| | BCLR #xx:3,@aa:32 | B | | | | | | 8 | | | | (#xx:3 of @aa:32)←0 | — | — | — | — | — | — | 6 |
| | BCLR Rn,Rd | B | | 2 | | | | | | | | (Rn8 of Rd8)←0 | — | — | — | — | — | — | 1 |
| | BCLR Rn,@ERd | B | | | 4 | | | | | | | (Rn8 of @ERd)←0 | — | — | — | — | — | — | 4 |
| | BCLR Rn,@aa:8 | B | | | | | | 4 | | | | (Rn8 of @aa:8)←0 | — | — | — | — | — | — | 4 |
| | BCLR Rn,@aa:16 | B | | | | | | 6 | | | | (Rn8 of @aa:16)←0 | — | — | — | — | — | — | 5 |

**Table A.1 Instruction Set (cont)**

**(5) Bit-Manipulation Instructions (cont)**

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | | Operation | Condition Code | | | | | | No. of States*1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | \| | | I | H | N | Z | V | C | Advanced |
| BCLR | BCLR Rn,@aa:32 | B | | | | | | 8 | | | | (Rn8 of @aa:32)←0 | — | — | — | — | — | — | 6 |
| BNOT | BNOT #xx:3,Rd | B | | 2 | | | | | | | | (#xx:3 of Rd8)←[¬ (#xx:3 of Rd8)] | — | — | — | — | — | — | 1 |
| | BNOT #xx:3,@ERd | B | | | 4 | | | | | | | (#xx:3 of @ERd)← [¬ (#xx:3 of @ERd)] | — | — | — | — | — | — | 4 |
| | BNOT #xx:3,@aa:8 | B | | | | | | 4 | | | | (#xx:3 of @aa:8)← [¬ (#xx:3 of @aa:8)] | — | — | — | — | — | — | 4 |
| | BNOT #xx:3,@aa:16 | B | | | | | | 6 | | | | (#xx:3 of @aa:16)← [¬ (#xx:3 of @aa:16)] | — | — | — | — | — | — | 5 |
| | BNOT #xx:3,@aa:32 | B | | | | | | 8 | | | | (#xx:3 of @aa:32)← [¬ (#xx:3 of @aa:32)] | — | — | — | — | — | — | 6 |
| | BNOT Rn,Rd | B | | 2 | | | | | | | | (Rn8 of Rd8)←[¬ (Rn8 of Rd8)] | — | — | — | — | — | — | 1 |
| | BNOT Rn,@ERd | B | | | 4 | | | | | | | (Rn8 of @ERd)←[¬ (Rn8 of @ERd)] | — | — | — | — | — | — | 4 |
| | BNOT Rn,@aa:8 | B | | | | | | 4 | | | | (Rn8 of @aa:8)←[¬ (Rn8 of @aa:8)] | — | — | — | — | — | — | 4 |
| | BNOT Rn,@aa:16 | B | | | | | | 6 | | | | (Rn8 of @aa:16)← [¬ (Rn8 of @aa:16)] | — | — | — | — | — | — | 5 |
| | BNOT Rn,@aa:32 | B | | | | | | 8 | | | | (Rn8 of @aa:32)← [¬ (Rn8 of @aa:32)] | — | — | — | — | — | — | 6 |
| BTST | BTST #xx:3,Rd | B | | 2 | | | | | | | | ¬ (#xx:3 of Rd8)→Z | — | — | — | ↕ | — | — | 1 |
| | BTST #xx:3,@ERd | B | | | 4 | | | | | | | ¬ (#xx:3 of @ERd)→Z | — | — | — | ↕ | — | — | 3 |
| | BTST #xx:3,@aa:8 | B | | | | | | 4 | | | | ¬ (#xx:3 of @aa:8)→Z | — | — | — | ↕ | — | — | 3 |
| | BTST #xx:3,@aa:16 | B | | | | | | 6 | | | | ¬ (#xx:3 of @aa:16)→Z | — | — | — | ↕ | — | — | 4 |



HITACHI

**Table A.1 Instruction Set (cont)**

**(5) Bit-Manipulation Instructions (cont)**

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | | Operation | Condition Code | | | | | | No. of States*1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | \| | | I | H | N | Z | V | C | Advanced |
| BTST | BTST #xx:3,@aa:32 | B | | | | | | 8 | | | | ¬ (#xx:3 of @aa:32)→Z | — | — | — | ↕ | — | — | 5 |
| | BTST Rn,Rd | B | | 2 | | | | | | | | ¬ (Rn8 of Rd8)→Z | — | — | — | ↕ | — | — | 1 |
| | BTST Rn,@ERd | B | | | 4 | | | | | | | ¬ (Rn8 of @ERd)→Z | — | — | — | ↕ | — | — | 3 |
| | BTST Rn,@aa:8 | B | | | | | | 4 | | | | ¬ (Rn8 of @aa:8)→Z | — | — | — | ↕ | — | — | 3 |
| | BTST Rn,@aa:16 | B | | | | | | 6 | | | | ¬ (Rn8 of @aa:16)→Z | — | — | — | ↕ | — | — | 4 |
| | BTST Rn,@aa:32 | B | | | | | | 8 | | | | ¬ (Rn8 of @aa:32)→Z | — | — | — | ↕ | — | — | 5 |
| BLD | BLD #xx:3,Rd | B | | 2 | | | | | | | | (#xx:3 of Rd8)→C | — | — | — | — | — | ↕ | 1 |
| | BLD #xx:3,@ERd | B | | | 4 | | | | | | | (#xx:3 of @ERd)→C | — | — | — | — | — | ↕ | 3 |
| | BLD #xx:3,@aa:8 | B | | | | | | 4 | | | | (#xx:3 of @aa:8)→C | — | — | — | — | — | ↕ | 3 |
| | BLD #xx:3,@aa:16 | B | | | | | | 6 | | | | (#xx:3 of @aa:16)→C | — | — | — | — | — | ↕ | 4 |
| | BLD #xx:3,@aa:32 | B | | | | | | 8 | | | | (#xx:3 of @aa:32)→C | — | — | — | — | — | ↕ | 5 |
| BILD | BILD #xx:3,Rd | B | | 2 | | | | | | | | ¬ (#xx:3 of Rd8)→C | — | — | — | — | — | ↕ | 1 |
| | BILD #xx:3,@ERd | B | | | 4 | | | | | | | ¬ (#xx:3 of @ERd)→C | — | — | — | — | — | ↕ | 3 |
| | BILD #xx:3,@aa:8 | B | | | | | | 4 | | | | ¬ (#xx:3 of @aa:8)→C | — | — | — | — | — | ↕ | 3 |
| | BILD #xx:3,@aa:16 | B | | | | | | 6 | | | | ¬ (#xx:3 of @aa:16)→C | — | — | — | — | — | ↕ | 4 |
| | BILD #xx:3,@aa:32 | B | | | | | | 8 | | | | ¬ (#xx:3 of @aa:32)→C | — | — | — | — | — | ↕ | 5 |
| BST | BST #xx:3,Rd | B | | 2 | | | | | | | | C→(#xx:3 of Rd8) | — | — | — | — | — | — | 1 |
| | BST #xx:3,@ERd | B | | | 4 | | | | | | | C→(#xx:3 of @ERd) | — | — | — | — | — | — | 4 |
| | BST #xx:3,@aa:8 | B | | | | | | 4 | | | | C→(#xx:3 of @aa:8) | — | — | — | — | — | — | 4 |

**Table A.1 Instruction Set (cont)**

**(5) Bit-Manipulation Instructions (cont)**

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | | Operation | Condition Code | | | | | | No. of States*[1] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | \| | | I | H | N | Z | V | C | Advanced |
| BST | BST #xx:3,@aa:16 | B | | | | | | 6 | | | | C→(#xx:3 of @aa:16) | — | — | — | — | — | — | 5 |
| | BST #xx:3,@aa:32 | B | | | | | | 8 | | | | C→(#xx:3 of @aa:32) | — | — | — | — | — | — | 6 |
| BIST | BIST #xx:3,Rd | B | | 2 | | | | | | | | ¬ C→(#xx:3 of Rd8) | — | — | — | — | — | — | 1 |
| | BIST #xx:3,@ERd | B | | | 4 | | | | | | | ¬ C→(#xx:3 of @ERd) | — | — | — | — | — | — | 4 |
| | BIST #xx:3,@aa:8 | B | | | | | | 4 | | | | ¬ C→(#xx:3 of @aa:8) | — | — | — | — | — | — | 4 |
| | BIST #xx:3,@aa:16 | B | | | | | | 6 | | | | ¬ C→(#xx:3 of @aa:16) | — | — | — | — | — | — | 5 |
| | BIST #xx:3,@aa:32 | B | | | | | | 8 | | | | ¬ C→(#xx:3 of @aa:32) | — | — | — | — | — | — | 6 |
| BAND | BAND #xx:3,Rd | B | | 2 | | | | | | | | C∧(#xx:3 of Rd8)→C | — | — | — | — | — | ↕ | 1 |
| | BAND #xx:3,@ERd | B | | | 4 | | | | | | | C∧(#xx:3 of @ERd)→C | — | — | — | — | — | ↕ | 3 |
| | BAND #xx:3,@aa:8 | B | | | | | | 4 | | | | C∧(#xx:3 of @aa:8)→C | — | — | — | — | — | ↕ | 3 |
| | BAND #xx:3,@aa:16 | B | | | | | | 6 | | | | C∧(#xx:3 of @aa:16)→C | — | — | — | — | — | ↕ | 4 |
| | BAND #xx:3,@aa:32 | B | | | | | | 8 | | | | C∧(#xx:3 of @aa:32)→C | — | — | — | — | — | ↕ | 5 |
| BIAND | BIAND #xx:3,Rd | B | | 2 | | | | | | | | C∧[¬ (#xx:3 of Rd8)]→C | — | — | — | — | — | ↕ | 1 |
| | BIAND #xx:3,@ERd | B | | | 4 | | | | | | | C∧[¬ (#xx:3 of @ERd)]→C | — | — | — | — | — | ↕ | 3 |
| | BIAND #xx:3,@aa:8 | B | | | | | | 4 | | | | C∧[¬ (#xx:3 of @aa:8)]→C | — | — | — | — | — | ↕ | 3 |
| | BIAND #xx:3,@aa:16 | B | | | | | | 6 | | | | C∧[¬ (#xx:3 of @aa:16)]→C | — | — | — | — | — | ↕ | 4 |
| | BIAND #xx:3,@aa:32 | B | | | | | | 8 | | | | C∧[¬ (#xx:3 of @aa:32)]→C | — | — | — | — | — | ↕ | 5 |
| BOR | BOR #xx:3,Rd | B | | 2 | | | | | | | | C∨(#xx:3 of Rd8)→C | — | — | — | — | — | ↕ | 1 |
| | BOR #xx:3,@ERd | B | | | 4 | | | | | | | C∨(#xx:3 of @ERd)→C | — | — | — | — | — | ↕ | 3 |



HITACHI

**Table A.1 Instruction Set (cont)**

**(5) Bit-Manipulation Instructions (cont)**

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | | Operation | Condition Code | | | | | | No. of States*1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | \| | | I | H | N | Z | V | C | Advanced |
| BOR | BOR #xx:3,@aa:8 | B | | | | | | 4 | | | | C∨(#xx:3 of @aa:8)→C | — | — | — | — | — | ↕ | 3 |
| | BOR #xx:3,@aa:16 | B | | | | | | 6 | | | | C∨(#xx:3 of @aa:16)→C | — | — | — | — | — | ↕ | 4 |
| | BOR #xx:3,@aa:32 | B | | | | | | 8 | | | | C∨(#xx:3 of @aa:32)→C | — | — | — | — | — | ↕ | 5 |
| BIOR | BIOR #xx:3,Rd | B | | 2 | | | | | | | | C∨[¬ (#xx:3 of Rd8)]→C | — | — | — | — | — | ↕ | 1 |
| | BIOR #xx:3,@ERd | B | | | 4 | | | | | | | C∨[¬ (#xx:3 of @ERd)]→C | — | — | — | — | — | ↕ | 3 |
| | BIOR #xx:3,@aa:8 | B | | | | | | 4 | | | | C∨[¬ (#xx:3 of @aa:8)]→C | — | — | — | — | — | ↕ | 3 |
| | BIOR #xx:3,@aa:16 | B | | | | | | 6 | | | | C∨[¬ (#xx:3 of @aa:16)]→C | — | — | — | — | — | ↕ | 4 |
| | BIOR #xx:3,@aa:32 | B | | | | | | 8 | | | | C∨[¬ (#xx:3 of @aa:32)]→C | — | — | — | — | — | ↕ | 5 |
| BXOR | BXOR #xx:3,Rd | B | | 2 | | | | | | | | C⊕(#xx:3 of Rd8)→C | — | — | — | — | — | ↕ | 1 |
| | BXOR #xx:3,@ERd | B | | | 4 | | | | | | | C⊕(#xx:3 of @ERd)→C | — | — | — | — | — | ↕ | 3 |
| | BXOR #xx:3,@aa:8 | B | | | | | | 4 | | | | C⊕(#xx:3 of @aa:8)→C | — | — | — | — | — | ↕ | 3 |
| | BXOR #xx:3,@aa:16 | B | | | | | | 6 | | | | C⊕(#xx:3 of @aa:16)→C | — | — | — | — | — | ↕ | 4 |
| | BXOR #xx:3,@aa:32 | B | | | | | | 8 | | | | C⊕(#xx:3 of @aa:32)→C | — | — | — | — | — | ↕ | 5 |
| BIXOR | BIXOR #xx:3,Rd | B | | 2 | | | | | | | | C⊕[¬ (#xx:3 of Rd8)]→C | — | — | — | — | — | ↕ | 1 |
| | BIXOR #xx:3,@ERd | B | | | 4 | | | | | | | C⊕[¬ (#xx:3 of @ERd)]→C | — | — | — | — | — | ↕ | 3 |
| | BIXOR #xx:3,@aa:8 | B | | | | | | 4 | | | | C⊕[¬ (#xx:3 of @aa:8)]→C | — | — | — | — | — | ↕ | 3 |
| | BIXOR #xx:3,@aa:16 | B | | | | | | 6 | | | | C⊕[¬ (#xx:3 of @aa:16)]→C | — | — | — | — | — | ↕ | 4 |
| | BIXOR #xx:3,@aa:32 | B | | | | | | 8 | | | | C⊕[¬ (#xx:3 of @aa:32)]→C | — | — | — | — | — | ↕ | 5 |

**Table A.1 Instruction Set**

**(6) Branch Instructions**

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | | Operation | Condition Code | | | | | | | No. of States*1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | — | | Branching Condition | I | H | N | Z | V | C | Advanced |
| Bcc | BRA d:8(BT d:8) | — | | | | | | | 2 | | | if condition is true then | Always | — | — | — | — | — | — | 2 |
| | BRA d:16(BT d:16) | — | | | | | | | 4 | | | PC←PC+d | | — | — | — | — | — | — | 3 |
| | BRN d:8(BF d:8) | — | | | | | | | 2 | | | else next; | Never | — | — | — | — | — | — | 2 |
| | BRN d:16(BF d:16) | — | | | | | | | 4 | | | | | — | — | — | — | — | — | 3 |
| | BHI d:8 | — | | | | | | | 2 | | | | C∨Z=0 | — | — | — | — | — | — | 2 |
| | BHI d:16 | — | | | | | | | 4 | | | | | — | — | — | — | — | — | 3 |
| | BLS d:8 | — | | | | | | | 2 | | | | C∨Z=1 | — | — | — | — | — | — | 2 |
| | BLS d:16 | — | | | | | | | 4 | | | | | — | — | — | — | — | — | 3 |
| | BCC d:B(BHS d:8) | — | | | | | | | 2 | | | | C=0 | — | — | — | — | — | — | 2 |
| | BCC d:16(BHS d:16) | — | | | | | | | 4 | | | | | — | — | — | — | — | — | 3 |
| | BCS d:8(BLO d:8) | — | | | | | | | 2 | | | | C=1 | — | — | — | — | — | — | 2 |
| | BCS d:16(BLO d:16) | — | | | | | | | 4 | | | | | — | — | — | — | — | — | 3 |
| | BNE d:8 | — | | | | | | | 2 | | | | Z=0 | — | — | — | — | — | — | 2 |
| | BNE d:16 | — | | | | | | | 4 | | | | | — | — | — | — | — | — | 3 |
| | BEQ d:8 | — | | | | | | | 2 | | | | Z=1 | — | — | — | — | — | — | 2 |
| | BEQ d:16 | — | | | | | | | 4 | | | | | — | — | — | — | — | — | 3 |
| | BVC d:8 | — | | | | | | | 2 | | | | V=0 | — | — | — | — | — | — | 2 |
| | BVC d:16 | — | | | | | | | 4 | | | | | — | — | — | — | — | — | 3 |

**Table A.1 Instruction Set (cont)**

**(6) Branch Instructions (cont)**

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | | Operation | Condition Code | | | | | | No. of States*[1] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | \| | Branching Condition | I | H | N | Z | V | C | Advanced |
| Bcc | BVS d:8 | — | | | | | | | 2 | | | V=1 | — | — | — | — | — | — | 2 |
| | BVS d:16 | — | | | | | | | 4 | | | | — | — | — | — | — | — | 3 |
| | BPL d:8 | — | | | | | | | 2 | | | N=0 | — | — | — | — | — | — | 2 |
| | BPL d:16 | — | | | | | | | 4 | | | | — | — | — | — | — | — | 3 |
| | BMI d:8 | — | | | | | | | 2 | | | N=1 | — | — | — | — | — | — | 2 |
| | BMI d:16 | — | | | | | | | 4 | | | | — | — | — | — | — | — | 3 |
| | BGE d:8 | — | | | | | | | 2 | | | N⊕V=0 | — | — | — | — | — | — | 2 |
| | BGE d:16 | — | | | | | | | 4 | | | | — | — | — | — | — | — | 3 |
| | BLT d:8 | — | | | | | | | 2 | | | N⊕V=1 | — | — | — | — | — | — | 2 |
| | BLT d:16 | — | | | | | | | 4 | | | | — | — | — | — | — | — | 3 |
| | BGT d:8 | — | | | | | | | 2 | | | Z∨(N⊕V)=0 | — | — | — | — | — | — | 2 |
| | BGT d:16 | — | | | | | | | 4 | | | | — | — | — | — | — | — | 3 |
| | BLE d:8 | — | | | | | | | 2 | | | Z∨(N⊕V)=1 | — | — | — | — | — | — | 2 |
| | BLE d:16 | — | | | | | | | 4 | | | | — | — | — | — | — | — | 3 |

**Table A.1 Instruction Set (cont)**

**(6) Branch Instructions (cont)**

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | | Operation | Condition Code | | | | | | No. of States*[1] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | \| | | I | H | N | Z | V | C | Advanced |
| JMP | JMP @ERn | — | | | 2 | | | | | | | PC←ERn | — | — | — | — | — | — | 2 |
| | JMP @aa:24 | — | | | | | | 4 | | | | PC←aa:24 | — | — | — | — | — | — | 3 |
| | JMP @@aa:8 | — | | | | | | | | 2 | | PC←@aa:8 | — | — | — | — | — | — | 5 |
| BSR | BSR d:8 | — | | | | | | | 2 | | | PC→@-SP,PC←PC+d:8 | — | — | — | — | — | — | 4 |
| | BSR d:16 | — | | | | | | | 4 | | | PC→@-SP,PC←PC+d:16 | — | — | — | — | — | — | 5 |
| JSR | JSR @ERn | — | | | 2 | | | | | | | PC→@-SP,PC←ERn | — | — | — | — | — | — | 4 |
| | JSR @aa:24 | — | | | | | | 4 | | | | PC→@-SP,PC←aa:24 | — | — | — | — | — | — | 5 |
| | JSR @@aa:8 | — | | | | | | | | 2 | | PC→@-SP,PC←@aa:8 | — | — | — | — | — | — | 6 |
| RTS | RTS | — | | | | | | | | | 2 | PC←@SP+ | — | — | — | — | — | — | 5 |

**Table A.1 Instruction Set**

**(7) System Control Instructions**

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | — | Operation | Condition Code I | H | N | Z | V | C | No. of States*1 Advanced |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TRAPA | TRAPA #xx:2 | — | | | | | | | | | | PC→@-SP,CCR→@-SP, EXR→@-SP,<vector>→PC | 1 | — | — | — | — | — | 8 [9] |
| RTE | RTE | — | | | | | | | | | | EXR←@SP+,CCR←@SP+, PC←@SP+ | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 5 [9] |
| SLEEP | SLEEP | — | | | | | | | | | | Transition to power-down state | — | — | — | — | — | — | 2 |
| LDC | LDC #xx:8,CCR | B | 2 | | | | | | | | | #xx:8→CCR | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 1 |
| | LDC #xx:8,EXR | B | 4 | | | | | | | | | #xx:8→EXR | — | — | — | — | — | — | 2 |
| | LDC Rs,CCR | B | | 2 | | | | | | | | Rs8→CCR | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 1 |
| | LDC Rs,EXR | B | | 2 | | | | | | | | Rs8→EXR | — | — | — | — | — | — | 1 |
| | LDC @ERs,CCR | W | | | 4 | | | | | | | @ERs→CCR | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 3 |
| | LDC @ERs,EXR | W | | | 4 | | | | | | | @ERs→EXR | — | — | — | — | — | — | 3 |
| | LDC @(d:16,ERs),CCR | W | | | | 6 | | | | | | @(d:16,ERs)→CCR | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 4 |
| | LDC @(d:16,ERs),EXR | W | | | | 6 | | | | | | @(d:16,ERs)→EXR | — | — | — | — | — | — | 4 |
| | LDC @(d:32,ERs),CCR | W | | | | 10 | | | | | | @(d:32,ERs)→CCR | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 6 |
| | LDC @(d:32,ERs),EXR | W | | | | 10 | | | | | | @(d:32,ERs)→EXR | — | — | — | — | — | — | 6 |
| | LDC @ERs+,CCR | W | | | | | 4 | | | | | @ERs→CCR,ERs32+2→ERs32 | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 4 |
| | LDC @ERs+,EXR | W | | | | | 4 | | | | | @ERs→EXR,ERs32+2→ERs32 | — | — | — | — | — | — | 4 |
| | LDC @aa:16,CCR | W | | | | | | 6 | | | | @aa:16→CCR | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 4 |
| | LDC @aa:16,EXR | W | | | | | | 6 | | | | @aa:16→EXR | — | — | — | — | — | — | 4 |
| | LDC @aa:32,CCR | W | | | | | | 8 | | | | @aa:32→CCR | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 5 |
| | LDC @aa:32,EXR | W | | | | | | 8 | | | | @aa:32→EXR | — | — | — | — | — | — | 5 |

**Table A.1 Instruction Set (cont)**

**(7) System Control Instructions (cont)**

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) | | | | | | | | | Operation | Condition Code | | | | | | No. of States*[1] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | \| | | I | H | N | Z | V | C | Advanced |
| STC | STC CCR,Rd | B | | 2 | | | | | | | | CCR→Rd8 | — | — | — | — | — | — | 1 |
| | STC EXR,Rd | B | | 2 | | | | | | | | EXR→Rd8 | — | — | — | — | — | — | 1 |
| | STC CCR,@ERd | W | | | 4 | | | | | | | CCR→@ERd | — | — | — | — | — | — | 3 |
| | STC EXR,@ERd | W | | | 4 | | | | | | | EXR→@ERd | — | — | — | — | — | — | 3 |
| | STC CCR,@(d:16,ERd) | W | | | | 6 | | | | | | CCR→@(d:16,ERd) | — | — | — | — | — | — | 4 |
| | STC EXR,@(d:16,ERd) | W | | | | 6 | | | | | | EXR→@(d:16,ERd) | — | — | — | — | — | — | 4 |
| | STC CCR,@(d:32,ERd) | W | | | | 10 | | | | | | CCR→@(d:32,ERd) | — | — | — | — | — | — | 6 |
| | STC EXR,@(d:32,ERd) | W | | | | 10 | | | | | | EXR→@(d:32,ERd) | — | — | — | — | — | — | 6 |
| | STC CCR,@-ERd | W | | | | | 4 | | | | | ERd32-2→ERd32,CCR→@ERd | — | — | — | — | — | — | 4 |
| | STC EXR,@-ERd | W | | | | | 4 | | | | | ERd32-2→ERd32,EXR→@ERd | — | — | — | — | — | — | 4 |
| | STC CCR,@aa:16 | W | | | | | | 6 | | | | CCR→@aa:16 | — | — | — | — | — | — | 4 |
| | STC EXR,@aa:16 | W | | | | | | 6 | | | | EXR→@aa:16 | — | — | — | — | — | — | 4 |
| | STC CCR,@aa:32 | W | | | | | | 8 | | | | CCR→@aa:32 | — | — | — | — | — | — | 5 |
| | STC EXR,@aa:32 | W | | | | | | 8 | | | | EXR→@aa:32 | — | — | — | — | — | — | 5 |
| ANDC | ANDC #xx:8,CCR | B | 2 | | | | | | | | | CCR∧#xx:8→CCR | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 1 |
| | ANDC #xx:8,EXR | B | 4 | | | | | | | | | EXR∧#xx:8→EXR | — | — | — | — | — | — | 2 |
| ORC | ORC #xx:8,CCR | B | 2 | | | | | | | | | CCR∨#xx:8→CCR | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 1 |
| | ORC #xx:8,EXR | B | 4 | | | | | | | | | EXR∨#xx:8→EXR | — | — | — | — | — | — | 2 |
| XORC | XORC #xx:8,CCR | B | 2 | | | | | | | | | CCR⊕#xx:8→CCR | ↕ | ↕ | ↕ | ↕ | ↕ | ↕ | 1 |
| | XORC #xx:8,EXR | B | 4 | | | | | | | | | EXR⊕#xx:8→EXR | — | — | — | — | — | — | 2 |
| NOP | NOP | — | | | | | | | | | 2 | PC←PC+2 | — | — | — | — | — | — | 1 |

## Table A.1 Instruction Set

### (8) Block Transfer Instructions

| Mnemonic | | Operand Size | Addressing Mode/ Instruction Length (Bytes) #xx | Rn | @ERn | @(d,ERn) | @–ERn/@ERn+ | @aa | @(d,PC) | @@aa | \| | Operation | Condition Code I | H | N | Z | V | C | No. of States*1 Advanced |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EEPMOV | EEPMOV.B | — | | | | | | | | | 4 | if R4L≠0<br>Repeat @ER5→@ER6<br>ER5+1→ER5<br>ER6+1→ER6<br>R4L-1→R4L<br>Until R4L=0<br>else next; | — | — | — | — | — | — | 4+2n *2 |
| | EEPMOV.W | — | | | | | | | | | 4 | if R4≠0<br>Repeat @ER5→@ER6<br>ER5+1→ER5<br>ER6+1→ER6<br>R4-1→R4<br>Until R4=0<br>else next; | — | — | — | — | — | — | 4+2n *2 |

Notes: 1. The number of states is the number of states required for execution when the instruction and its operands are located in on-chip memory.
2. n is the initial value of R4L or R4.
[1] Seven states for saving or restoring two registers, nine states for three registers, or eleven states for four registers.
[2] Cannot be used in the H8S/2345 Series.
[3] Set to 1 when a carry or borrow occurs at bit 11; otherwise cleared to 0.
[4] Set to 1 when a carry or borrow occurs at bit 27; otherwise cleared to 0.
[5] Retains its previous value when the result is zero; otherwise cleared to 0.
[6] Set to 1 when the divisor is negative; otherwise cleared to 0.
[7] Set to 1 when the divisor is zero; otherwise cleared to 0.
[8] Set to 1 when the quotient is negative; otherwise cleared to 0.
[9] One additional state is required for execution when EXR is valid.

# A.2 Instruction Codes

Table A.2 shows the instruction codes.

**Table A.2 Instruction Codes**

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | 2nd byte | 3rd byte | 4th byte | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte | |
| ADD | ADD.B #xx:8,Rd | B | 8 rd | IMM | | | | | | | | | |
| | ADD.B Rs,Rd | B | 0 8 | rs rd | | | | | | | | | |
| | ADD.W #xx:16,Rd | W | 7 9 | 1 rd | IMM | | | | | | | | |
| | ADD.W Rs,Rd | W | 0 9 | rs rd | | | | | | | | | |
| | ADD.L #xx:32,ERd | L | 7 A | 1 0 erd | IMM | | | | | | | | |
| | ADD.L ERs,ERd | L | 0 A | 1 ers 0 erd | | | | | | | | | |
| ADDS | ADDS #1,ERd | L | 0 B | 0 0 erd | | | | | | | | | |
| | ADDS #2,ERd | L | 0 B | 8 0 erd | | | | | | | | | |
| | ADDS #4,ERd | L | 0 B | 9 0 erd | | | | | | | | | |
| ADDX | ADDX #xx:8,Rd | B | 9 rd | IMM | | | | | | | | | |
| | ADDX Rs,Rd | B | 0 E | rs rd | | | | | | | | | |
| AND | AND.B #xx:8,Rd | B | E rd | IMM | | | | | | | | | |
| | AND.B Rs,Rd | B | 1 6 | rs rd | | | | | | | | | |
| | AND.W #xx:16,Rd | W | 7 9 | 6 rd | IMM | | | | | | | | |
| | AND.W Rs,Rd | W | 6 6 | rs rd | | | | | | | | | |
| | AND.L #xx:32,ERd | L | 7 A | 6 0 erd | IMM | | | | | | | | |
| | AND.L ERs,ERd | L | 0 1 | F 0 | 6 6 | 0 ers 0 erd | | | | | | | |
| ANDC | ANDC #xx:8,CCR | B | 0 6 | IMM | | | | | | | | | |
| | ANDC #xx:8,EXR | B | 0 1 | 4 1 | 0 6 | IMM | | | | | | | |
| BAND | BAND #xx:3,Rd | B | 7 6 | 0 IMM rd | | | | | | | | | |
| | BAND #xx:3,@ERd | B | 7 C | 0 erd 0 | 7 6 | 0 IMM 0 | | | | | | | |
| | BAND #xx:3,@aa:8 | B | 7 E | abs | 7 6 | 0 IMM 0 | | | | | | | |
| | BAND #xx:3,@aa:16 | B | 6 A | 1 0 | abs | | 7 6 | 0 IMM 0 | | | | | |
| | BAND #xx:3,@aa:32 | B | 6 A | 3 0 | abs | | | | 7 6 | 0 IMM 0 | | | |
| Bcc | BRA d:8 (BT d:8) | — | 4 0 | disp | | | | | | | | | |
| | BRA d:16 (BT d:16) | — | 5 8 | 0 0 | disp | | | | | | | | |
| | BRN d:8 (BF d:8) | — | 4 1 | disp | | | | | | | | | |
| | BRN d:16 (BF d:16) | — | 5 8 | 1 0 | disp | | | | | | | | |

## Table A.2 Instruction Codes (cont)

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | 2nd byte | 3rd byte | 4th byte | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte |
| Bcc | BHI d:8 | — | 4 2 | disp | | | | | | | | |
| | BHI d:16 | — | 5 8 | 2 0 | disp | | | | | | | |
| | BLS d:8 | — | 4 3 | disp | | | | | | | | |
| | BLS d:16 | — | 5 8 | 3 0 | disp | | | | | | | |
| | BCC d:8 (BHS d:8) | — | 4 4 | disp | | | | | | | | |
| | BCC d:16 (BHS d:16) | — | 5 8 | 4 0 | disp | | | | | | | |
| | BCS d:8 (BLO d:8) | — | 4 5 | disp | | | | | | | | |
| | BCS d:16 (BLO d:16) | — | 5 8 | 5 0 | disp | | | | | | | |
| | BNE d:8 | — | 4 6 | disp | | | | | | | | |
| | BNE d:16 | — | 5 8 | 6 0 | disp | | | | | | | |
| | BEQ d:8 | — | 4 7 | disp | | | | | | | | |
| | BEQ d:16 | — | 5 8 | 7 0 | disp | | | | | | | |
| | BVC d:8 | — | 4 8 | disp | | | | | | | | |
| | BVC d:16 | — | 5 8 | 8 0 | disp | | | | | | | |
| | BVS d:8 | — | 4 9 | disp | | | | | | | | |
| | BVS d:16 | — | 5 8 | 9 0 | disp | | | | | | | |
| | BPL d:8 | — | 4 A | disp | | | | | | | | |
| | BPL d:16 | — | 5 8 | A 0 | disp | | | | | | | |
| | BMI d:8 | — | 4 B | disp | | | | | | | | |
| | BMI d:16 | — | 5 8 | B 0 | disp | | | | | | | |
| | BGE d:8 | — | 4 C | disp | | | | | | | | |
| | BGE d:16 | — | 5 8 | C 0 | disp | | | | | | | |
| | BLT d:8 | — | 4 D | disp | | | | | | | | |
| | BLT d:16 | — | 5 8 | D 0 | disp | | | | | | | |
| | BGT d:8 | — | 4 E | disp | | | | | | | | |
| | BGT d:16 | — | 5 8 | E 0 | disp | | | | | | | |
| | BLE d:8 | — | 4 F | disp | | | | | | | | |
| | BLE d:16 | — | 5 8 | F 0 | disp | | | | | | | |

## Table A.2 Instruction Codes (cont)

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | | 5th byte | | 6th byte | | 7th byte | | 8th byte | | 9th byte | 10th byte |
| BCLR | BCLR #xx:3,Rd | B | 7 | 2 | 0 IMM | rd | | | | | | | | | | | | | | |
| | BCLR #xx:3,@ERd | B | 7 | D | 0 erd | 0 | 7 | 2 | 0 IMM | 0 | | | | | | | | | | |
| | BCLR #xx:3,@aa:8 | B | 7 | F | abs | | 7 | 2 | 0 IMM | 0 | | | | | | | | | | |
| | BCLR #xx:3,@aa:16 | B | 6 | A | 1 | 8 | abs | | | | 7 | 2 | 0 IMM | 0 | | | | | | |
| | BCLR #xx:3,@aa:32 | B | 6 | A | 3 | 8 | abs | | | | | | | | 7 | 2 | 0 IMM | 0 | | |
| | BCLR Rn,Rd | B | 6 | 2 | rn | rd | | | | | | | | | | | | | | |
| | BCLR Rn,@ERd | B | 7 | D | 0 erd | 0 | 6 | 2 | rn | 0 | | | | | | | | | | |
| | BCLR Rn,@aa:8 | B | 7 | F | abs | | 6 | 2 | rn | 0 | | | | | | | | | | |
| | BCLR Rn,@aa:16 | B | 6 | A | 1 | 8 | abs | | | | 6 | 2 | rn | 0 | | | | | | |
| | BCLR Rn,@aa:32 | B | 6 | A | 3 | 8 | abs | | | | | | | | 6 | 2 | rn | 0 | | |
| BIAND | BIAND #xx:3,Rd | B | 7 | 6 | 1 IMM | rd | | | | | | | | | | | | | | |
| | BIAND #xx:3,@ERd | B | 7 | C | 0 erd | 0 | 7 | 6 | 1 IMM | 0 | | | | | | | | | | |
| | BIAND #xx:3,@aa:8 | B | 7 | E | abs | | 7 | 6 | 1 IMM | 0 | | | | | | | | | | |
| | BIAND #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | 7 | 6 | 1 IMM | 0 | | | | | | |
| | BIAND #xx:3,@aa:32 | B | 6 | A | 3 | 0 | abs | | | | | | | | 7 | 6 | 1 IMM | 0 | | |
| BILD | BILD #xx:3,Rd | B | 7 | 7 | 1 IMM | rd | | | | | | | | | | | | | | |
| | BILD #xx:3,@ERd | B | 7 | C | 0 erd | 0 | 7 | 7 | 1 IMM | 0 | | | | | | | | | | |
| | BILD #xx:3,@aa:8 | B | 7 | E | abs | | 7 | 7 | 1 IMM | 0 | | | | | | | | | | |
| | BILD #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | 7 | 7 | 1 IMM | 0 | | | | | | |
| | BILD #xx:3,@aa:32 | B | 6 | A | 3 | 0 | abs | | | | | | | | 7 | 7 | 1 IMM | 0 | | |
| BIOR | BIOR #xx:3,Rd | B | 7 | 4 | 1 IMM | rd | | | | | | | | | | | | | | |
| | BIOR #xx:3,@ERd | B | 7 | C | 0 erd | 0 | 7 | 4 | 1 IMM | 0 | | | | | | | | | | |
| | BIOR #xx:3,@aa:8 | B | 7 | E | abs | | 7 | 4 | 1 IMM | 0 | | | | | | | | | | |
| | BIOR #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | 7 | 4 | 1 IMM | 0 | | | | | | |
| | BIOR #xx:3,@aa:32 | B | 6 | A | 3 | 0 | abs | | | | | | | | 7 | 4 | 1 IMM | 0 | | |

## Table A.2 Instruction Codes (cont)

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | | 5th byte | | 6th byte | | 7th byte | | 8th byte | | 9th byte | 10th byte |
| BIST | BIST #xx:3,Rd | B | 6 | 7 | 1 | IMM | rd | | | | | | | | | | | | | |
| | BIST #xx:3,@ERd | B | 7 | D | 0 | erd | 0 | 6 | 7 | 1 | IMM | 0 | | | | | | | | |
| | BIST #xx:3,@aa:8 | B | 7 | F | abs | | | 6 | 7 | 1 | IMM | 0 | | | | | | | | |
| | BIST #xx:3,@aa:16 | B | 6 | A | 1 | 8 | abs | | | | | | 6 | 7 | 1 | IMM | 0 | | | |
| | BIST #xx:3,@aa:32 | B | 6 | A | 3 | 8 | abs | | | | | | | | 6 | 7 | 1 | IMM | 0 | |
| BIXOR | BIXOR #xx:3,Rd | B | 7 | 5 | 1 | IMM | rd | | | | | | | | | | | | | |
| | BIXOR #xx:3,@ERd | B | 7 | C | 0 | erd | 0 | 7 | 5 | 1 | IMM | 0 | | | | | | | | |
| | BIXOR #xx:3,@aa:8 | B | 7 | E | abs | | | 7 | 5 | 1 | IMM | 0 | | | | | | | | |
| | BIXOR #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | | | 7 | 5 | 1 | IMM | 0 | | | |
| | BIXOR #xx:3,@aa:32 | B | 6 | A | 3 | 0 | abs | | | | | | | | 7 | 5 | 1 | IMM | 0 | |
| BLD | BLD #xx:3,Rd | B | 7 | 7 | 0 | IMM | rd | | | | | | | | | | | | | |
| | BLD #xx:3,@ERd | B | 7 | C | 0 | erd | 0 | 7 | 7 | 0 | IMM | 0 | | | | | | | | |
| | BLD #xx:3,@aa:8 | B | 7 | E | abs | | | 7 | 7 | 0 | IMM | 0 | | | | | | | | |
| | BLD #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | | | 7 | 7 | 0 | IMM | 0 | | | |
| | BLD #xx:3,@aa:32 | B | 6 | A | 3 | 0 | abs | | | | | | | | 7 | 7 | 0 | IMM | 0 | |
| BNOT | BNOT #xx:3,Rd | B | 7 | 1 | 0 | IMM | rd | | | | | | | | | | | | | |
| | BNOT #xx:3,@ERd | B | 7 | D | 0 | erd | 0 | 7 | 1 | 0 | IMM | 0 | | | | | | | | |
| | BNOT #xx:3,@aa:8 | B | 7 | F | abs | | | 7 | 1 | 0 | IMM | 0 | | | | | | | | |
| | BNOT #xx:3,@aa:16 | B | 6 | A | 1 | 8 | abs | | | | | | 7 | 1 | 0 | IMM | 0 | | | |
| | BNOT #xx:3,@aa:32 | B | 6 | A | 3 | 8 | abs | | | | | | | | 7 | 1 | 0 | IMM | 0 | |
| | BNOT Rn,Rd | B | 6 | 1 | rn | rd | | | | | | | | | | | | | | |
| | BNOT Rn,@ERd | B | 7 | D | 0 | erd | 0 | 6 | 1 | rn | | 0 | | | | | | | | |
| | BNOT Rn,@aa:8 | B | 7 | F | abs | | | 6 | 1 | rn | | 0 | | | | | | | | |
| | BNOT Rn,@aa:16 | B | 6 | A | 1 | 8 | abs | | | | | | 6 | 1 | rn | | 0 | | | |
| | BNOT Rn,@aa:32 | B | 6 | A | 3 | 8 | abs | | | | | | | | 6 | 1 | rn | | 0 | |

## Table A.2 Instruction Codes (cont)

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | | 5th byte | | 6th byte | | 7th byte | | 8th byte | | 9th byte | 10th byte |
| BOR | BOR #xx:3,Rd | B | 7 | 4 | 0 IMM | rd | | | | | | | | | | | | | | |
| | BOR #xx:3,@ERd | B | 7 | C | 0 erd | 0 | 7 | 4 | 0 IMM | 0 | | | | | | | | | | |
| | BOR #xx:3,@aa:8 | B | 7 | E | abs | | 7 | 4 | 0 IMM | 0 | | | | | | | | | | |
| | BOR #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | 7 | 4 | 0 IMM | 0 | | | | | | |
| | BOR #xx:3,@aa:32 | B | 6 | A | 3 | 0 | abs | | | | | | | | 7 | 4 | 0 IMM | 0 | | |
| BSET | BSET #xx:3,Rd | B | 7 | 0 | 0 IMM | rd | | | | | | | | | | | | | | |
| | BSET #xx:3,@ERd | B | 7 | D | 0 erd | 0 | 7 | 0 | 0 IMM | 0 | | | | | | | | | | |
| | BSET #xx:3,@aa:8 | B | 7 | F | abs | | 7 | 0 | 0 IMM | 0 | | | | | | | | | | |
| | BSET #xx:3,@aa:16 | B | 6 | A | 1 | 8 | abs | | | | 7 | 0 | 0 IMM | 0 | | | | | | |
| | BSET #xx:3,@aa:32 | B | 6 | A | 3 | 8 | abs | | | | | | | | 7 | 0 | 0 IMM | 0 | | |
| | BSET Rn,Rd | B | 6 | 0 | rn | rd | | | | | | | | | | | | | | |
| | BSET Rn,@ERd | B | 7 | D | 0 erd | 0 | 6 | 0 | rn | 0 | | | | | | | | | | |
| | BSET Rn,@aa:8 | B | 7 | F | abs | | 6 | 0 | rn | 0 | | | | | | | | | | |
| | BSET Rn,@aa:16 | B | 6 | A | 1 | 8 | abs | | | | 6 | 0 | rn | 0 | | | | | | |
| | BSET Rn,@aa:32 | B | 6 | A | 3 | 8 | abs | | | | | | | | 6 | 0 | rn | 0 | | |
| BSR | BSR d:8 | — | 5 | 5 | disp | | | | | | | | | | | | | | | |
| | BSR d:16 | — | 5 | C | 0 | 0 | disp | | | | | | | | | | | | | |
| BST | BST #xx:3,Rd | B | 6 | 7 | 0 IMM | rd | | | | | | | | | | | | | | |
| | BST #xx:3,@ERd | B | 7 | D | 0 erd | 0 | 6 | 7 | 0 IMM | 0 | | | | | | | | | | |
| | BST #xx:3,@aa:8 | B | 7 | F | abs | | 6 | 7 | 0 IMM | 0 | | | | | | | | | | |
| | BST #xx:3,@aa:16 | B | 6 | A | 1 | 8 | abs | | | | 6 | 7 | 0 IMM | 0 | | | | | | |
| | BST #xx:3,@aa:32 | B | 6 | A | 3 | 8 | abs | | | | | | | | 6 | 7 | 0 IMM | 0 | | |
| BTST | BTST #xx:3,Rd | B | 7 | 3 | 0 IMM | rd | | | | | | | | | | | | | | |
| | BTST #xx:3,@ERd | B | 7 | C | 0 erd | 0 | 7 | 3 | 0 IMM | 0 | | | | | | | | | | |
| | BTST #xx:3,@aa:8 | B | 7 | E | abs | | 7 | 3 | 0 IMM | 0 | | | | | | | | | | |
| | BTST #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | 7 | 3 | 0 IMM | 0 | | | | | | |
| | BTST #xx:3,@aa:32 | B | 6 | A | 3 | 0 | abs | | | | | | | | 7 | 3 | 0 IMM | 0 | | |
| | BTST Rn,Rd | B | 6 | 3 | rn | rd | | | | | | | | | | | | | | |
| | BTST Rn,@ERd | B | 7 | C | 0 erd | 0 | 6 | 3 | rn | 0 | | | | | | | | | | |

## Table A.2 Instruction Codes (cont)

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | | 5th byte | | 6th byte | | 7th byte | | 8th byte | | 9th byte | 10th byte |
| BTST | BTST Rn,@aa:8 | B | 7 | E | abs | | 6 | 3 | rn | 0 | | | | | | | | | | |
| | BTST Rn,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | 6 | 3 | rn | 0 | | | | | | |
| | BTST Rn,@aa:32 | B | 6 | A | 3 | 0 | abs | | | | | | | | 6 | 3 | rn | 0 | | |
| BXOR | BXOR #xx:3,Rd | B | 7 | 5 | 0 | IMM | rd | | | | | | | | | | | | | |
| | BXOR #xx:3,@ERd | B | 7 | C | 0 | erd | 0 | 7 | 5 | 0 | IMM | 0 | | | | | | | | |
| | BXOR #xx:3,@aa:8 | B | 7 | E | abs | | 7 | 5 | 0 | IMM | 0 | | | | | | | | | |
| | BXOR #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | 7 | 5 | 0 | IMM | 0 | | | | | |
| | BXOR #xx:3,@aa:32 | B | 6 | A | 3 | 0 | abs | | | | | | | | 7 | 5 | 0 | IMM | 0 | |
| CLRMAC | CLRMAC | — | Cannot be used in the H8S/2345 Series | | | | | | | | | | | | | | | | | |
| CMP | CMP.B #xx:8,Rd | B | A | rd | IMM | | | | | | | | | | | | | | | |
| | CMP.B Rs,Rd | B | 1 | C | rs | rd | | | | | | | | | | | | | | |
| | CMP.W #xx:16,Rd | W | 7 | 9 | 2 | rd | IMM | | | | | | | | | | | | | |
| | CMP.W Rs,Rd | W | 1 | D | rs | rd | | | | | | | | | | | | | | |
| | CMP.L #xx:32,ERd | L | 7 | A | 2 | 0 | erd | IMM | | | | | | | | | | | | |
| | CMP.L ERs,ERd | L | 1 | F | 1 | ers | 0 | erd | | | | | | | | | | | | |
| DAA | DAA Rd | B | 0 | F | 0 | rd | | | | | | | | | | | | | | |
| DAS | DAS Rd | B | 1 | F | 0 | rd | | | | | | | | | | | | | | |
| DEC | DEC.B Rd | B | 1 | A | 0 | rd | | | | | | | | | | | | | | |
| | DEC.W #1,Rd | W | 1 | B | 5 | rd | | | | | | | | | | | | | | |
| | DEC.W #2,Rd | W | 1 | B | D | rd | | | | | | | | | | | | | | |
| | DEC.L #1,ERd | L | 1 | B | 7 | 0 | erd | | | | | | | | | | | | | |
| | DEC.L #2,ERd | L | 1 | B | F | 0 | erd | | | | | | | | | | | | | |
| DIVXS | DIVXS.B Rs,Rd | B | 0 | 1 | D | 0 | 5 | 1 | rs | rd | | | | | | | | | | |
| | DIVXS.W Rs,ERd | W | 0 | 1 | D | 0 | 5 | 3 | rs | 0 | erd | | | | | | | | | |
| DIVXU | DIVXU.B Rs,Rd | B | 5 | 1 | rs | rd | | | | | | | | | | | | | | |
| | DIVXU.W Rs,ERd | W | 5 | 3 | rs | 0 | erd | | | | | | | | | | | | | |
| EEPMOV | EEPMOV.B | — | 7 | B | 5 | C | 5 | 9 | 8 | F | | | | | | | | | | |
| | EEPMOV.W | — | 7 | B | D | 4 | 5 | 9 | 8 | F | | | | | | | | | | |

## Table A.2 Instruction Codes (cont)

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | | 5th byte | | 6th byte | | 7th byte | | 8th byte | | 9th byte | 10th byte |
| EXTS | EXTS.W Rd | W | 1 | 7 | D | rd | | | | | | | | | | | | | | |
| | EXTS.L ERd | L | 1 | 7 | F | 0 erd | | | | | | | | | | | | | | |
| EXTU | EXTU.W Rd | W | 1 | 7 | 5 | rd | | | | | | | | | | | | | | |
| | EXTU.L ERd | L | 1 | 7 | 7 | 0 erd | | | | | | | | | | | | | | |
| INC | INC.B Rd | B | 0 | A | 0 | rd | | | | | | | | | | | | | | |
| | INC.W #1,Rd | W | 0 | B | 5 | rd | | | | | | | | | | | | | | |
| | INC.W #2,Rd | W | 0 | B | D | rd | | | | | | | | | | | | | | |
| | INC.L #1,ERd | L | 0 | B | 7 | 0 erd | | | | | | | | | | | | | | |
| | INC.L #2,ERd | L | 0 | B | F | 0 erd | | | | | | | | | | | | | | |
| JMP | JMP @ERn | — | 5 | 9 | 0 ern | 0 | | | | | | | | | | | | | | |
| | JMP @aa:24 | — | 5 | A | abs | | | | | | | | | | | | | | | |
| | JMP @@aa:8 | — | 5 | B | abs | | | | | | | | | | | | | | | |
| JSR | JSR @ERn | — | 5 | D | 0 ern | 0 | | | | | | | | | | | | | | |
| | JSR @aa:24 | — | 5 | E | abs | | | | | | | | | | | | | | | |
| | JSR @@aa:8 | — | 5 | F | abs | | | | | | | | | | | | | | | |
| LDC | LDC #xx:8,CCR | B | 0 | 7 | IMM | | | | | | | | | | | | | | | |
| | LDC #xx:8,EXR | B | 0 | 1 | 4 | 1 | 0 | 7 | IMM | | | | | | | | | | | |
| | LDC Rs,CCR | B | 0 | 3 | 0 | rs | | | | | | | | | | | | | | |
| | LDC Rs,EXR | B | 0 | 3 | 1 | rs | | | | | | | | | | | | | | |
| | LDC @ERs,CCR | W | 0 | 1 | 4 | 0 | 6 | 9 | 0 ers | 0 | | | | | | | | | | |
| | LDC @ERs,EXR | W | 0 | 1 | 4 | 1 | 6 | 9 | 0 ers | 0 | | | | | | | | | | |
| | LDC @(d:16,ERs),CCR | W | 0 | 1 | 4 | 0 | 6 | F | 0 ers | 0 | disp | | | | | | | | | |
| | LDC @(d:16,ERs),EXR | W | 0 | 1 | 4 | 1 | 6 | F | 0 ers | 0 | disp | | | | | | | | | |
| | LDC @(d:32,ERs),CCR | W | 0 | 1 | 4 | 0 | 7 | 8 | 0 ers | 0 | 6 | B | 2 | 0 | disp | | | | | |
| | LDC @(d:32,ERs),EXR | W | 0 | 1 | 4 | 1 | 7 | 8 | 0 ers | 0 | 6 | B | 2 | 0 | disp | | | | | |
| | LDC @ERs+,CCR | W | 0 | 1 | 4 | 0 | 6 | D | 0 ers | 0 | | | | | | | | | | |
| | LDC @ERs+,EXR | W | 0 | 1 | 4 | 1 | 6 | D | 0 ers | 0 | | | | | | | | | | |
| | LDC @aa:16,CCR | W | 0 | 1 | 4 | 0 | 6 | B | 0 | 0 | disp | | | | | | | | | |
| | LDC @aa:16,EXR | W | 0 | 1 | 4 | 1 | 6 | B | 0 | 0 | disp | | | | | | | | | |

## Table A.2 Instruction Codes (cont)

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte |
| LDC | LDC @aa:32,CCR | W | 0 | 1 | 4 | 0 | 6 | B | 2 | 0 | abs | | | | | |
| | LDC @aa:32,EXR | W | 0 | 1 | 4 | 1 | 6 | B | 2 | 0 | abs | | | | | |
| LDM | LDM.L @SP+, (ERn-ERn+1) | L | 0 | 1 | 1 | 0 | 6 | D | 7 | 0 ern+1 | | | | | | |
| | LDM.L @SP+, (ERn-ERn+2) | L | 0 | 1 | 2 | 0 | 6 | D | 7 | 0 ern+2 | | | | | | |
| | LDM.L @SP+, (ERn-ERn+3) | L | 0 | 1 | 3 | 0 | 6 | D | 7 | 0 ern+3 | | | | | | |
| LDMAC | LDMAC ERs,MACH | L | Cannot be used in the H8S/2345 Series | | | | | | | | | | | | | |
| | LDMAC ERs,MACL | L | | | | | | | | | | | | | | |
| MAC | MAC @ERn+,@ERm+ | — | | | | | | | | | | | | | | |
| MOV | MOV.B #xx:8,Rd | B | F | rd | IMM | | | | | | | | | | | |
| | MOV.B Rs,Rd | B | 0 | C | rs | rd | | | | | | | | | | |
| | MOV.B @ERs,Rd | B | 6 | 8 | 0 ers | rd | | | | | | | | | | |
| | MOV.B @(d:16,ERs),Rd | B | 6 | E | 0 ers | rd | disp | | | | | | | | | |
| | MOV.B @(d:32,ERs),Rd | B | 7 | 8 | 0 ers | 0 | 6 | A | 2 | rd | disp | | | | | |
| | MOV.B @ERs+,Rd | B | 6 | C | 0 ers | rd | | | | | | | | | | |
| | MOV.B @aa:8,Rd | B | 2 | rd | abs | | | | | | | | | | | |
| | MOV.B @aa:16,Rd | B | 6 | A | 0 | rd | abs | | | | | | | | | |
| | MOV.B @aa:32,Rd | B | 6 | A | 2 | rd | abs | | | | | | | | | |
| | MOV.B Rs,@ERd | B | 6 | 8 | 1 erd | rs | | | | | | | | | | |
| | MOV.B Rs,@(d:16,ERd) | B | 6 | E | 1 erd | rs | disp | | | | | | | | | |
| | MOV.B Rs,@(d:32,ERd) | B | 7 | 8 | 0 erd | 0 | 6 | A | A | rs | disp | | | | | |
| | MOV.B Rs,@-ERd | B | 6 | C | 1 erd | rs | | | | | | | | | | |
| | MOV.B Rs,@aa:8 | B | 3 | rs | abs | | | | | | | | | | | |
| | MOV.B Rs,@aa :16 | B | 6 | A | 8 | rs | abs | | | | | | | | | |
| | MOV.B Rs,@aa:32 | B | 6 | A | A | rs | abs | | | | | | | | | |
| | MOV.W #xx:16,Rd | W | 7 | 9 | 0 | rd | IMM | | | | | | | | | |
| | MOV.W Rs,Rd | W | 0 | D | rs | rd | | | | | | | | | | |
| | MOV.W @ERs,Rd | W | 6 | 9 | 0 ers | rd | | | | | | | | | | |
| | MOV.W @(d:16,ERs),Rd | W | 6 | F | 0 ers | rd | disp | | | | | | | | | |
| | MOV.W @(d:32,ERs),Rd | W | 7 | 8 | 0 ers | 0 | 6 | B | 2 | rd | disp | | | | | |

## Table A.2 Instruction Codes (cont)

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | 2nd byte | 3rd byte | 4th byte | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte |
| MOV | MOV.W @ERs+,Rd | W | 6 D | 0 ers rd | | | | | | | | |
| | MOV.W @aa:16,Rd | W | 6 B | 0 rd | abs | | | | | | | |
| | MOV.W @aa:32,Rd | W | 6 B | 2 rd | abs | | | | | | | |
| | MOV.W Rs,@ERd | W | 6 9 | 1 erd rs | | | | | | | | |
| | MOV.W Rs,@(d:16,ERd) | W | 6 F | 1 erd rs | disp | | | | | | | |
| | MOV.W Rs,@(d:32,ERd) | W | 7 8 | 0 erd 0 | 6 B | A rs | disp | | | | | |
| | MOV.W Rs,@-ERd | W | 6 D | 1 erd rs | | | | | | | | |
| | MOV.W Rs,@aa:16 | W | 6 B | 8 rs | abs | | | | | | | |
| | MOV.W Rs,@aa:32 | W | 6 B | A rs | abs | | | | | | | |
| | MOV.L #xx:32,Rd | L | 7 A | 0 0 erd | IMM | | | | | | | |
| | MOV.L ERs,ERd | L | 0 F | 1 ers 0 erd | | | | | | | | |
| | MOV.L @ERs,ERd | L | 0 1 | 0 0 | 6 9 | 0 ers 0 erd | | | | | | |
| | MOV.L @(d:16,ERs),ERd | L | 0 1 | 0 0 | 6 F | 0 ers 0 erd | disp | | | | | |
| | MOV.L @(d:32,ERs),ERd | L | 0 1 | 0 0 | 7 8 | 0 ers 0 | 6 B | 2 0 erd | disp | | | |
| | MOV.L @ERs+,ERd | L | 0 1 | 0 0 | 6 D | 0 ers 0 erd | | | | | | |
| | MOV.L @aa:16 ,ERd | L | 0 1 | 0 0 | 6 B | 0 0 erd | abs | | | | | |
| | MOV.L @aa:32 ,ERd | L | 0 1 | 0 0 | 6 B | 2 0 erd | abs | | | | | |
| | MOV.L ERs,@ERd | L | 0 1 | 0 0 | 6 9 | 1 erd 0 ers | | | | | | |
| | MOV.L ERs,@(d:16,ERd) | L | 0 1 | 0 0 | 6 F | 1 erd 0 ers | disp | | | | | |
| | MOV.L ERs,@(d:32,ERd)* | L | 0 1 | 0 0 | 7 8 | 0 erd 0 | 6 B | A 0 ers | disp | | | |
| | MOV.L ERs,@-ERd | L | 0 1 | 0 0 | 6 D | 1 erd 0 ers | | | | | | |
| | MOV.L ERs,@aa:16 | L | 0 1 | 0 0 | 6 B | 8 0 ers | abs | | | | | |
| | MOV.L ERs,@aa:32 | L | 0 1 | 0 0 | 6 B | A 0 ers | abs | | | | | |
| MOVFPE | MOVFPE @aa:16,Rd | B | Cannot be used in the H8S/2345 Series | | | | | | | | | |
| MOVTPE | MOVTPE Rs,@aa:16 | B | | | | | | | | | | |
| MULXS | MULXS.B Rs,Rd | B | 0 1 | C 0 | 5 0 | rs rd | | | | | | |
| | MULXS.W Rs,ERd | W | 0 1 | C 0 | 5 2 | rs 0 erd | | | | | | |
| MULXU | MULXU.B Rs,Rd | B | 5 0 | rs rd | | | | | | | | |
| | MULXU.W Rs,ERd | W | 5 2 | rs 0 erd | | | | | | | | |

## Table A.2 Instruction Codes (cont)

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | | 2nd byte | | 3rd byte | 4th byte | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte |
| NEG | NEG.B Rd | B | 1 | 7 | 8 | rd | | | | | | | | |
| | NEG.W Rd | W | 1 | 7 | 9 | rd | | | | | | | | |
| | NEG.L ERd | L | 1 | 7 | B | 0 erd | | | | | | | | |
| NOP | NOP | — | 0 | 0 | 0 | 0 | | | | | | | | |
| NOT | NOT.B Rd | B | 1 | 7 | 0 | rd | | | | | | | | |
| | NOT.W Rd | W | 1 | 7 | 1 | rd | | | | | | | | |
| | NOT.L ERd | L | 1 | 7 | 3 | 0 erd | | | | | | | | |
| OR | OR.B #xx:8,Rd | B | C | rd | IMM | | | | | | | | | |
| | OR.B Rs,Rd | B | 1 | 4 | rs | rd | | | | | | | | |
| | OR.W #xx:16,Rd | W | 7 | 9 | 4 | rd | IMM | | | | | | | |
| | OR.W Rs,Rd | W | 6 | 4 | rs | rd | | | | | | | | |
| | OR.L #xx:32,ERd | L | 7 | A | 4 | 0 erd | IMM | | | | | | | |
| | OR.L ERs,ERd | L | 0 | 1 | F | 0 | 6 4 | 0 ers 0 erd | | | | | | |
| ORC | ORC #xx:8,CCR | B | 0 | 4 | IMM | | | | | | | | | |
| | ORC #xx:8,EXR | B | 0 | 1 | 4 | 1 | 0 4 | IMM | | | | | | |
| POP | POP.W Rn | W | 6 | D | 7 | rn | | | | | | | | |
| | POP.L ERn | L | 0 | 1 | 0 | 0 | 6 D | 7 0 ern | | | | | | |
| PUSH | PUSH.W Rn | W | 6 | D | F | rn | | | | | | | | |
| | PUSH.L ERn | L | 0 | 1 | 0 | 0 | 6 D | F 0 ern | | | | | | |
| ROTL | ROTL.B Rd | B | 1 | 2 | 8 | rd | | | | | | | | |
| | ROTL.B #2, Rd | B | 1 | 2 | C | rd | | | | | | | | |
| | ROTL.W Rd | W | 1 | 2 | 9 | rd | | | | | | | | |
| | ROTL.W #2, Rd | W | 1 | 2 | D | rd | | | | | | | | |
| | ROTL.L ERd | L | 1 | 2 | B | 0 erd | | | | | | | | |
| | ROTL.L #2, ERd | L | 1 | 2 | F | 0 erd | | | | | | | | |

## Table A.2 Instruction Codes (cont)

| Instruc-tion | Mnemonic | Size | Instruction Format | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | | 2nd byte | | 3rd byte | 4th byte | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte |
| ROTR | ROTR.B Rd | B | 1 | 3 | 8 | rd | | | | | | | | |
| | ROTR.B #2, Rd | B | 1 | 3 | C | rd | | | | | | | | |
| | ROTR.W Rd | W | 1 | 3 | 9 | rd | | | | | | | | |
| | ROTR.W #2, Rd | W | 1 | 3 | D | rd | | | | | | | | |
| | ROTR.L ERd | L | 1 | 3 | B | 0 erd | | | | | | | | |
| | ROTR.L #2, ERd | L | 1 | 3 | F | 0 erd | | | | | | | | |
| ROTXL | ROTXL.B Rd | B | 1 | 2 | 0 | rd | | | | | | | | |
| | ROTXL.B #2, Rd | B | 1 | 2 | 4 | rd | | | | | | | | |
| | ROTXL.W Rd | W | 1 | 2 | 1 | rd | | | | | | | | |
| | ROTXL.W #2, Rd | W | 1 | 2 | 5 | rd | | | | | | | | |
| | ROTXL.L ERd | L | 1 | 2 | 3 | 0 erd | | | | | | | | |
| | ROTXL.L #2, ERd | L | 1 | 2 | 7 | 0 erd | | | | | | | | |
| ROTXR | ROTXR.B Rd | B | 1 | 3 | 0 | rd | | | | | | | | |
| | ROTXR.B #2, Rd | B | 1 | 3 | 4 | rd | | | | | | | | |
| | ROTXR.W Rd | W | 1 | 3 | 1 | rd | | | | | | | | |
| | ROTXR.W #2, Rd | W | 1 | 3 | 5 | rd | | | | | | | | |
| | ROTXR.L ERd | L | 1 | 3 | 3 | 0 erd | | | | | | | | |
| | ROTXR.L #2, ERd | L | 1 | 3 | 7 | 0 erd | | | | | | | | |
| RTE | RTE | — | 5 | 6 | 7 | 0 | | | | | | | | |
| RTS | RTS | — | 5 | 4 | 7 | 0 | | | | | | | | |
| SHAL | SHAL.B Rd | B | 1 | 0 | 8 | rd | | | | | | | | |
| | SHAL.B #2, Rd | B | 1 | 0 | C | rd | | | | | | | | |
| | SHAL.W Rd | W | 1 | 0 | 9 | rd | | | | | | | | |
| | SHAL.W #2, Rd | W | 1 | 0 | D | rd | | | | | | | | |
| | SHAL.L ERd | L | 1 | 0 | B | 0 erd | | | | | | | | |
| | SHAL.L #2, ERd | L | 1 | 0 | F | 0 erd | | | | | | | | |

## Table A.2 Instruction Codes (cont)

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | | | 5th byte | | | | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte |
| SHAR | SHAR.B Rd | B | 1 | 1 | 8 | rd | | | | | | | | | | | | | | |
| | SHAR.B #2, Rd | B | 1 | 1 | C | rd | | | | | | | | | | | | | | |
| | SHAR.W Rd | W | 1 | 1 | 9 | rd | | | | | | | | | | | | | | |
| | SHAR.W #2, Rd | W | 1 | 1 | D | rd | | | | | | | | | | | | | | |
| | SHAR.L ERd | L | 1 | 1 | B | 0 erd | | | | | | | | | | | | | | |
| | SHAR.L #2, ERd | L | 1 | 1 | F | 0 erd | | | | | | | | | | | | | | |
| SHLL | SHLL.B Rd | B | 1 | 0 | 0 | rd | | | | | | | | | | | | | | |
| | SHLL.B #2, Rd | B | 1 | 0 | 4 | rd | | | | | | | | | | | | | | |
| | SHLL.W Rd | W | 1 | 0 | 1 | rd | | | | | | | | | | | | | | |
| | SHLL.W #2, Rd | W | 1 | 0 | 5 | rd | | | | | | | | | | | | | | |
| | SHLL.L ERd | L | 1 | 0 | 3 | 0 erd | | | | | | | | | | | | | | |
| | SHLL.L #2, ERd | L | 1 | 0 | 7 | 0 erd | | | | | | | | | | | | | | |
| SHLR | SHLR.B Rd | B | 1 | 1 | 0 | rd | | | | | | | | | | | | | | |
| | SHLR.B #2, Rd | B | 1 | 1 | 4 | rd | | | | | | | | | | | | | | |
| | SHLR.W Rd | W | 1 | 1 | 1 | rd | | | | | | | | | | | | | | |
| | SHLR.W #2, Rd | W | 1 | 1 | 5 | rd | | | | | | | | | | | | | | |
| | SHLR.L ERd | L | 1 | 1 | 3 | 0 erd | | | | | | | | | | | | | | |
| | SHLR.L #2, ERd | L | 1 | 1 | 7 | 0 erd | | | | | | | | | | | | | | |
| SLEEP | SLEEP | — | 0 | 1 | 8 | 0 | | | | | | | | | | | | | | |
| STC | STC.B CCR,Rd | B | 0 | 2 | 0 | rd | | | | | | | | | | | | | | |
| | STC.B EXR,Rd | B | 0 | 2 | 1 | rd | | | | | | | | | | | | | | |
| | STC.W CCR,@ERd | W | 0 | 1 | 4 | 0 | 6 | 9 | 1 erd | | 0 | | | | | | | | | |
| | STC.W EXR,@ERd | W | 0 | 1 | 4 | 1 | 6 | 9 | 1 erd | | 0 | | | | | | | | | |
| | STC.W CCR,@(d:16,ERd) | W | 0 | 1 | 4 | 0 | 6 | F | 1 erd | | 0 | disp | | | | | | | | |
| | STC.W EXR,@(d:16,ERd) | W | 0 | 1 | 4 | 1 | 6 | F | 1 erd | | 0 | disp | | | | | | | | |
| | STC.W CCR,@(d:32,ERd) | W | 0 | 1 | 4 | 0 | 7 | 8 | 0 erd | | 0 | 6 | B | A | 0 | disp | | | | |
| | STC.W EXR,@(d:32,ERd) | W | 0 | 1 | 4 | 1 | 7 | 8 | 0 erd | | 0 | 6 | B | A | 0 | disp | | | | |
| | STC.W CCR,@-ERd | W | 0 | 1 | 4 | 0 | 6 | D | 1 erd | | 0 | | | | | | | | | |
| | STC.W EXR,@-ERd | W | 0 | 1 | 4 | 1 | 6 | D | 1 erd | | 0 | | | | | | | | | |

## Table A.2 Instruction Codes (cont)

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte | | |
| STC | STC.W CCR,@aa:16 | W | 0 | 1 | 4 | 0 | 6 | B | 8 | 0 | abs | | | | | | | |
| | STC.W EXR,@aa:16 | W | 0 | 1 | 4 | 1 | 6 | B | 8 | 0 | abs | | | | | | | |
| | STC.W CCR,@aa:32 | W | 0 | 1 | 4 | 0 | 6 | B | A | 0 | abs | | | | | | | |
| | STC.W EXR,@aa:32 | W | 0 | 1 | 4 | 1 | 6 | B | A | 0 | abs | | | | | | | |
| STM | STM.L(ERn-ERn+1), @-SP | L | 0 | 1 | 1 | 0 | 6 | D | F | 0 ern | | | | | | | | |
| | STM.L (ERn-ERn+2), @-SP | L | 0 | 1 | 2 | 0 | 6 | D | F | 0 ern | | | | | | | | |
| | STM.L (ERn-ERn+3), @-SP | L | 0 | 1 | 3 | 0 | 6 | D | F | 0 ern | | | | | | | | |
| STMAC | STMAC MACH,ERd | L | Cannot be used in the H8S/2345 Series | | | | | | | | | | | | | | | |
| | STMAC MACL,ERd | L | | | | | | | | | | | | | | | | |
| SUB | SUB.B Rs,Rd | B | 1 | 8 | rs | rd | | | | | | | | | | | | |
| | SUB.W #xx:16,Rd | W | 7 | 9 | 3 | rd | IMM | | | | | | | | | | | |
| | SUB.W Rs,Rd | W | 1 | 9 | rs | rd | | | | | | | | | | | | |
| | SUB.L #xx:32,ERd | L | 7 | A | 3 | 0 erd | IMM | | | | | | | | | | | |
| | SUB.L ERs,ERd | L | 1 | A | 1 ers | 0 erd | | | | | | | | | | | | |
| SUBS | SUBS #1,ERd | L | 1 | B | 0 | 0 erd | | | | | | | | | | | | |
| | SUBS #2,ERd | L | 1 | B | 8 | 0 erd | | | | | | | | | | | | |
| | SUBS #4,ERd | L | 1 | B | 9 | 0 erd | | | | | | | | | | | | |
| SUBX | SUBX #xx:8,Rd | B | B | rd | IMM | | | | | | | | | | | | | |
| | SUBX Rs,Rd | B | 1 | E | rs | rd | | | | | | | | | | | | |
| TAS | TAS @ERd | B | 0 | 1 | E | 0 | 7 | B | 0 erd | C | | | | | | | | |
| TRAPA | TRAPA #x:2 | — | 5 | 7 | 00 IMM | 0 | | | | | | | | | | | | |
| XOR | XOR.B #xx:8,Rd | B | D | rd | IMM | | | | | | | | | | | | | |
| | XOR.B Rs,Rd | B | 1 | 5 | rs | rd | | | | | | | | | | | | |
| | XOR.W #xx:16,Rd | W | 7 | 9 | 5 | rd | IMM | | | | | | | | | | | |
| | XOR.W Rs,Rd | W | 6 | 5 | rs | rd | | | | | | | | | | | | |
| | XOR.L #xx:32,ERd | L | 7 | A | 5 | 0 erd | IMM | | | | | | | | | | | |
| | XOR.L ERs,ERd | L | 0 | 1 | F | 0 | 6 | 5 | 0 ers | 0 erd | | | | | | | | |

## Table A.2 Instruction Codes (cont)

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1st byte | 2nd byte | 3rd byte | 4th byte | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte |
| XORC | XORC #xx:8,CCR | B | 0 5 | IMM | | | | | | | | |
| | XORC #xx:8,EXR | B | 0 1 | 4 1 | 0 5 | IMM | | | | | | |

Note: * Bit 7 of the 4th byte of the MOV.L ERs, @(d:32,ERd) instruction can be either 1 or 0.

**Legend**

| | |
|---|---|
| IMM: | Immediate data (2, 3, 8, 16, or 32 bits) |
| abs: | Absolute address (8, 16, 24, or 32 bits) |
| disp: | Displacement (8, 16, or 32 bits) |
| rs, rd, rn: | Register field (4 bits specifying an 8-bit or 16-bit register. The symbols rs, rd, and rn correspond to operand symbols Rs, Rd,and Rn.) |
| ers, erd, ern, erm: | Register field (3 bits specifying an address register or 32-bit register. The symbols ers, erd, ern, and erm correspond to operand symbols ERs, ERd, ERn, and ERm.) |

The register fields specify general registers as follows.

| Address Register 32-Bit Register | | 16-Bit Register | | 8-Bit Register | |
|---|---|---|---|---|---|
| Register Field | General Register | Register Field | General Register | Register Field | General Register |
| 000 | ER0 | 0000 | R0 | 0000 | R0H |
| 001 | ER1 | 0001 | R1 | 0001 | R1H |
| • | • | • | • | • | • |
| • | • | • | • | • | • |
| • | • | • | • | • | • |
| 111 | ER7 | 0111 | R7 | 0111 | R7H |
| | | 1000 | E0 | 1000 | R0L |
| | | 1001 | E1 | 1001 | R1L |
| | | • | • | • | • |
| | | • | • | • | • |
| | | • | • | • | • |
| | | 1111 | E7 | 1111 | R7L |

# A.3 Operation Code Map

Table A.3 shows the operation code map.

**Table A.3 Operation Code Map (1)**




| AH \ AL | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | NOP | Table A.3(2) | STC / * STMAC | LDC / * LDMAC | ORC | XORC | ANDC | LDC | ADD | | Table A.3(2) | Table A.3(2) | MOV | | ADDX | Table A.3(2) |
| 1 | Table A.3(2) | Table A.3(2) | Table A.3(2) | Table A.3(2) | OR | XOR | AND | Table A.3(2) | SUB | | Table A.3(2) | Table A.3(2) | CMP | | SUBX | Table A.3(2) |
| 2 | MOV.B | | | | | | | | | | | | | | | |
| 3 | | | | | | | | | | | | | | | | |
| 4 | BRA | BRN | BHI | BLS | BCC | BCS | BNE | BEQ | BVC | BVS | BPL | BMI | BGE | BLT | BGT | BLE |
| 5 | MULXU | DIVXU | MULXU | DIVXU | RTS | BSR | RTE | TRAPA | Table A.3(2) | JMP | | | BSR | JSR | | |
| 6 | BSET | BNOT | BCLR | BTST | OR / BIOR | XOR / BXOR / BIXOR | AND / BAND / BIAND | BST / BIST / BLD / BILD | MOV | | Table A.3(2) | | MOV | | | |
| 7 | | | | | | | | | MOV | Table A.3(2) | Table A.3(2) | EEPMOV | Table A.3(3) | | | |
| 8 | ADD | | | | | | | | | | | | | | | |
| 9 | ADDX | | | | | | | | | | | | | | | |
| A | CMP | | | | | | | | | | | | | | | |
| B | SUBX | | | | | | | | | | | | | | | |
| C | OR | | | | | | | | | | | | | | | |
| D | XOR | | | | | | | | | | | | | | | |
| E | AND | | | | | | | | | | | | | | | |
| F | MOV | | | | | | | | | | | | | | | |

Note: * Cannot be used in the H8S/2345 Series.

## Table A.3 Operation Code Map (2)

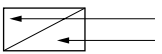
| Instruction code | 1st byte | | 2nd byte | |
|---|---|---|---|---|
| | AH | AL | BH | BL |

| AH AL \ BH | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 01 | MOV | LDM / STM | | | LDC / STC | | MAC* | | SLEEP | | CLRMAC* | | Table A.3(3) | Table A.3(3) | TAS | Table A.3(3) |
| 0A | INC | | | | | | | | ADD | | | | | | | |
| 0B | ADDS | | | | | INC | | INC | ADDS | | | | | INC | | INC |
| 0F | DAA | | | | | | | | MOV | | | | | | | |
| 10 | SHLL | | | SHLL | | | | SHLL | SHAL | | | SHAL | | | | SHAL |
| 11 | SHLR | | | SHLR | | | | SHLR | SHAR | | | SHAR | | | | SHAR |
| 12 | ROTXL | | | ROTXL | | | | ROTXL | ROTL | | | ROTL | | | | ROTL |
| 13 | ROTXR | | | ROTXR | | | | ROTXR | ROTR | | | ROTR | | | | ROTR |
| 17 | NOT | | | NOT | | EXTU | | EXTU | NEG | | | NEG | | EXTS | | EXTS |
| 1A | DEC | | | | | | | | SUB | | | | | | | |
| 1B | SUBS | | | | | DEC | | DEC | SUBS | | | | | DEC | | DEC |
| 1F | DAS | | | | | | | | CMP | | | | | | | |
| 58 | BRA | BRN | BHI | BLS | BCC | BCS | BNE | BEQ | BVC | BVS | BPL | BMI | BGE | BLT | BGT | BLE |
| 6A | MOV | Table A.3(4) | MOV | Table A.3(4) | MOVFPE* | | | | MOV | | MOV | | MOVTPE* | | | |
| 79 | MOV | ADD | CMP | SUB | OR | XOR | AND | | | | | | | | | |
| 7A | MOV | ADD | CMP | SUB | OR | XOR | AND | | | | | | | | | |

Note: * Cannot be used in the H8S/2345 Series.

## Table A.3 Operation Code Map (3)

| Instruction code | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | |
|---|---|---|---|---|---|---|---|---|
| | AH | AL | BH | BL | CH | CL | DH | DL |



Instruction when most significant bit of DH is 0.
Instruction when most significant bit of DH is 1.

| AH AL BH BL CH \ CL | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 01C05 | MULXS | | MULXS | | | | | | | | | | | | | |
| 01D05 | | DIVXS | | DIVXS | | | | | | | | | | | | |
| 01F06 | | | | | OR | XOR | AND | | | | | | | | | |
| 7Cr06 *1 | | | | BTST | | | | | | | | | | | | |
| 7Cr07 *1 | | | | BTST | BOR / BIOR | BXOR / BIXOR | BAND / BIAND | BLD / BILD | | | | | | | | |
| 7Dr06 *1 | BSET | BNOT | BCLR | | | | | BST / BIST | | | | | | | | |
| 7Dr07 *1 | BSET | BNOT | BCLR | | | | | | | | | | | | | |
| 7Eaa6 *2 | | | | BTST | | | | | | | | | | | | |
| 7Eaa7 *2 | | | | BTST | BOR / BIOR | BXOR / BIXOR | BAND / BIAND | BLD / BILD | | | | | | | | |
| 7Faa6 *2 | BSET | BNOT | BCLR | | | | | BST / BIST | | | | | | | | |
| 7Faa7 *2 | BSET | BNOT | BCLR | | | | | | | | | | | | | |

Notes: 1. r is the register specification field.
2. aa is the absolute address specification.

## Table A.3 Operation Code Map (4)

| Instruction code | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | | 5th byte | | 6th byte | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AH | AL | BH | BL | CH | CL | DH | DL | EH | EL | FH | FL |

Instruction when most significant bit of FH is 0.
Instruction when most significant bit of FH is 1.

| EL / AHALBHBLCHCLDHDLEH | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 6A10aaaa6* | | | | BTST | | | | | | | | | | | | |
| 6A10aaaa7* | | | | | BOR / BIOR | BXOR / BIXOR | BAND / BIAND | BLD / BILD | | | | | | | | |
| 6A18aaaa6* | BSET | BNOT | BCLR | | | | | BST / BIST | | | | | | | | |
| 6A18aaaa7* | | | | | | | | | | | | | | | | |

| Instruction code | 1st byte | | 2nd byte | | 3rd byte | | 4th byte | | 5th byte | | 6th byte | | 7th byte | | 8th byte | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AH | AL | BH | BL | CH | CL | DH | DL | EH | EL | FH | FL | GH | GL | HH | HL |

Instruction when most significant bit of HH is 0.
Instruction when most significant bit of HH is 1.

| GL / AHALBHBL ... FHFLGH | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 6A30aaaaaaaa6* | | | | BTST | | | | | | | | | | | | |
| 6A30aaaaaaaa7* | | | | | BOR / BIOR | BXOR / BIXOR | BAND / BIAND | BLD / BILD | | | | | | | | |
| 6A38aaaaaaaa6* | BSET | BNOT | BCLR | | | | | BST / BIST | | | | | | | | |
| 6A38aaaaaaaa7* | | | | | | | | | | | | | | | | |

Note: * aa is the absolute address specification.

## A.4 Number of States Required for Instruction Execution

The tables in this section can be used to calculate the number of states required for instruction execution by the CPU. Table A.5 indicates the number of instruction fetch, data read/write, and other cycles occurring in each instruction. Table A.4 indicates the number of states required for each cycle. The number of states required for execution of an instruction can be calculated from these two tables as follows:

$$\text{Execution states} = I \times S_I + J \times S_J + K \times S_K + L \times S_L + M \times S_M + N \times S_N$$

**Examples:** Advanced mode, program code and stack located in external memory, on-chip supporting modules accessed in two states with 8-bit bus width, external devices accessed in three states with one wait state and 16-bit bus width.

1. BSET #0, @FFFFC7:8
   From table A.5:

   $I = L = 2, \quad J = K = M = N = 0$

   From table A.4:

   $S_I = 4, \quad S_L = 2$

   Number of states required for execution = $2 \times 4 + 2 \times 2 = 12$

2. JSR @@30
   From table A.5:

   $I = J = K = 2, \quad L = M = N = 0$

   From table A.4:

   $S_I = S_J = S_K = 4$

   Number of states required for execution = $2 \times 4 + 2 \times 4 + 2 \times 4 = 24$

**Table A.4 Number of States per Cycle**

| | | Access Conditions | | | | | |
|---|---|---|---|---|---|---|---|
| | | On-Chip Supporting Module | | External Device | | | |
| | | | | 8-Bit Bus | | 16-Bit Bus | |
| Cycle | | On-Chip Memory | 8-Bit Bus | 16-Bit Bus | 2-State Access | 3-State Access | 2-State Access | 3-State Access |
| Instruction fetch | $S_I$ | 1 | 4 | 2 | 4 | 6 + 2m | 2 | 3 + m |
| Branch address read | $S_J$ | | | | | | | |
| Stack operation | $S_K$ | | | | | | | |
| Byte data access | $S_L$ | | 2 | | 2 | 3 + m | | |
| Word data access | $S_M$ | | 4 | | 4 | 6 + 2m | | |
| Internal operation | $S_N$ | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Legend

m: Number of wait states inserted into external device access

## Table A.5 Number of Cycles in Instruction Execution

| Instruction | Mnemonic | Instruction Fetch I | Branch Address Read J | Stack Operation K | Byte Data Access L | Word Data Access M | Internal Operation N |
|---|---|---|---|---|---|---|---|
| ADD | ADD.B #xx:8,Rd | 1 | | | | | |
| | ADD.B Rs,Rd | 1 | | | | | |
| | ADD.W #xx:16,Rd | 2 | | | | | |
| | ADD.W Rs,Rd | 1 | | | | | |
| | ADD.L #xx:32,ERd | 3 | | | | | |
| | ADD.L ERs,ERd | 1 | | | | | |
| ADDS | ADDS #1/2/4,ERd | 1 | | | | | |
| ADDX | ADDX #xx:8,Rd | 1 | | | | | |
| | ADDX Rs,Rd | 1 | | | | | |
| AND | AND.B #xx:8,Rd | 1 | | | | | |
| | AND.B Rs,Rd | 1 | | | | | |
| | AND.W #xx:16,Rd | 2 | | | | | |
| | AND.W Rs,Rd | 1 | | | | | |
| | AND.L #xx:32,ERd | 3 | | | | | |
| | AND.L ERs,ERd | 2 | | | | | |
| ANDC | ANDC #xx:8,CCR | 1 | | | | | |
| | ANDC #xx:8,EXR | 2 | | | | | |
| BAND | BAND #xx:3,Rd | 1 | | | | | |
| | BAND #xx:3,@ERd | 2 | | | 1 | | |
| | BAND #xx:3,@aa:8 | 2 | | | 1 | | |
| | BAND #xx:3,@aa:16 | 3 | | | 1 | | |
| | BAND #xx:3,@aa:32 | 4 | | | 1 | | |
| Bcc | BRA d:8 (BT d:8) | 2 | | | | | |
| | BRN d:8 (BF d:8) | 2 | | | | | |
| | BHI d:8 | 2 | | | | | |
| | BLS d:8 | 2 | | | | | |
| | BCC d:8 (BHS d:8) | 2 | | | | | |
| | BCS d:8 (BLO d:8) | 2 | | | | | |
| | BNE d:8 | 2 | | | | | |
| | BEQ d:8 | 2 | | | | | |
| | BVC d:8 | 2 | | | | | |
| | BVS d:8 | 2 | | | | | |
| | BPL d:8 | 2 | | | | | |

**Table A.5 Number of Cycles in Instruction Execution (cont)**

| Instruction | Mnemonic | Instruction Fetch I | Branch Address Read J | Stack Operation K | Byte Data Access L | Word Data Access M | Internal Operation N |
|---|---|---|---|---|---|---|---|
| Bcc | BMI d:8 | 2 | | | | | |
| | BGE d:8 | 2 | | | | | |
| | BLT d:8 | 2 | | | | | |
| | BGT d:8 | 2 | | | | | |
| | BLE d:8 | 2 | | | | | |
| | BRA d:16 (BT d:16) | 2 | | | | | 1 |
| | BRN d:16 (BF d:16) | 2 | | | | | 1 |
| | BHI d:16 | 2 | | | | | 1 |
| | BLS d:16 | 2 | | | | | 1 |
| | BCC d:16 (BHS d:16) | 2 | | | | | 1 |
| | BCS d:16 (BLO d:16) | 2 | | | | | 1 |
| | BNE d:16 | 2 | | | | | 1 |
| | BEQ d:16 | 2 | | | | | 1 |
| | BVC d:16 | 2 | | | | | 1 |
| | BVS d:16 | 2 | | | | | 1 |
| | BPL d:16 | 2 | | | | | 1 |
| | BMI d:16 | 2 | | | | | 1 |
| | BGE d:16 | 2 | | | | | 1 |
| | BLT d:16 | 2 | | | | | 1 |
| | BGT d:16 | 2 | | | | | 1 |
| | BLE d:16 | 2 | | | | | 1 |
| BCLR | BCLR #xx:3,Rd | 1 | | | | | |
| | BCLR #xx:3,@ERd | 2 | | | 2 | | |
| | BCLR #xx:3,@aa:8 | 2 | | | 2 | | |
| | BCLR #xx:3,@aa:16 | 3 | | | 2 | | |
| | BCLR #xx:3,@aa:32 | 4 | | | 2 | | |
| | BCLR Rn,Rd | 1 | | | | | |
| | BCLR Rn,@ERd | 2 | | | 2 | | |
| | BCLR Rn,@aa:8 | 2 | | | 2 | | |
| | BCLR Rn,@aa:16 | 3 | | | 2 | | |
| | BCLR Rn,@aa:32 | 4 | | | 2 | | |

**Table A.5 Number of Cycles in Instruction Execution (cont)**

| Instruction | Mnemonic | Instruction Fetch I | Branch Address Read J | Stack Operation K | Byte Data Access L | Word Data Access M | Internal Operation N |
|---|---|---|---|---|---|---|---|
| BIAND | BIAND #xx:3,Rd | 1 | | | | | |
| | BIAND #xx:3,@ERd | 2 | | | 1 | | |
| | BIAND #xx:3,@aa:8 | 2 | | | 1 | | |
| | BIAND #xx:3,@aa:16 | 3 | | | 1 | | |
| | BIAND #xx:3,@aa:32 | 4 | | | 1 | | |
| BILD | BILD #xx:3,Rd | 1 | | | | | |
| | BILD #xx:3,@ERd | 2 | | | 1 | | |
| | BILD #xx:3,@aa:8 | 2 | | | 1 | | |
| | BILD #xx:3,@aa:16 | 3 | | | 1 | | |
| | BILD #xx:3,@aa:32 | 4 | | | 1 | | |
| BIOR | BIOR #xx:8,Rd | 1 | | | | | |
| | BIOR #xx:8,@ERd | 2 | | | 1 | | |
| | BIOR #xx:8,@aa:8 | 2 | | | 1 | | |
| | BIOR #xx:8,@aa:16 | 3 | | | 1 | | |
| | BIOR #xx:8,@aa:32 | 4 | | | 1 | | |
| BIST | BIST #xx:3,Rd | 1 | | | | | |
| | BIST #xx:3,@ERd | 2 | | | 2 | | |
| | BIST #xx:3,@aa:8 | 2 | | | 2 | | |
| | BIST #xx:3,@aa:16 | 3 | | | 2 | | |
| | BIST #xx:3,@aa:32 | 4 | | | 2 | | |
| BIXOR | BIXOR #xx:3,Rd | 1 | | | | | |
| | BIXOR #xx:3,@ERd | 2 | | | 1 | | |
| | BIXOR #xx:3,@aa:8 | 2 | | | 1 | | |
| | BIXOR #xx:3,@aa:16 | 3 | | | 1 | | |
| | BIXOR #xx:3,@aa:32 | 4 | | | 1 | | |
| BLD | BLD #xx:3,Rd | 1 | | | | | |
| | BLD #xx:3,@ERd | 2 | | | 1 | | |
| | BLD #xx:3,@aa:8 | 2 | | | 1 | | |
| | BLD #xx:3,@aa:16 | 3 | | | 1 | | |
| | BLD #xx:3,@aa:32 | 4 | | | 1 | | |

**Table A.5 Number of Cycles in Instruction Execution (cont)**

| Instruction | Mnemonic | Instruction Fetch | Branch Address Read | Stack Operation | Byte Data Access | Word Data Access | Internal Operation |
|---|---|---|---|---|---|---|---|
| | | I | J | K | L | M | N |
| BNOT | BNOT #xx:3,Rd | 1 | | | | | |
| | BNOT #xx:3,@ERd | 2 | | | 2 | | |
| | BNOT #xx:3,@aa:8 | 2 | | | 2 | | |
| | BNOT #xx:3,@aa:16 | 3 | | | 2 | | |
| | BNOT #xx:3,@aa:32 | 4 | | | 2 | | |
| | BNOT Rn,Rd | 1 | | | | | |
| | BNOT Rn,@ERd | 2 | | | 2 | | |
| | BNOT Rn,@aa:8 | 2 | | | 2 | | |
| | BNOT Rn,@aa:16 | 3 | | | 2 | | |
| | BNOT Rn,@aa:32 | 4 | | | 2 | | |
| BOR | BOR #xx:3,Rd | 1 | | | | | |
| | BOR #xx:3,@ERd | 2 | | | 1 | | |
| | BOR #xx:3,@aa:8 | 2 | | | 1 | | |
| | BOR #xx:3,@aa:16 | 3 | | | 1 | | |
| | BOR #xx:3,@aa:32 | 4 | | | 1 | | |
| BSET | BSET #xx:3,Rd | 1 | | | | | |
| | BSET #xx:3,@ERd | 2 | | | 2 | | |
| | BSET #xx:3,@aa:8 | 2 | | | 2 | | |
| | BSET #xx:3,@aa:16 | 3 | | | 2 | | |
| | BSET #xx:3,@aa:32 | 4 | | | 2 | | |
| | BSET Rn,Rd | 1 | | | | | |
| | BSET Rn,@ERd | 2 | | | 2 | | |
| | BSET Rn,@aa:8 | 2 | | | 2 | | |
| | BSET Rn,@aa:16 | 3 | | | 2 | | |
| | BSET Rn,@aa:32 | 4 | | | 2 | | |
| BSR | BSR d:8 | 2 | | 2 | | | |
| | BSR d:16 | 2 | | 2 | | | 1 |
| BST | BST #xx:3,Rd | 1 | | | | | |
| | BST #xx:3,@ERd | 2 | | | 2 | | |
| | BST #xx:3,@aa:8 | 2 | | | 2 | | |
| | BST #xx:3,@aa:16 | 3 | | | 2 | | |
| | BST #xx:3,@aa:32 | 4 | | | 2 | | |

**Table A.5 Number of Cycles in Instruction Execution (cont)**

| Instruction | Mnemonic | Instruction Fetch | Branch Address Read | Stack Operation | Byte Data Access | Word Data Access | Internal Operation |
|---|---|---|---|---|---|---|---|
| | | I | J | K | L | M | N |
| BTST | BTST #xx:3,Rd | 1 | | | | | |
| | BTST #xx:3,@ERd | 2 | | | 1 | | |
| | BTST #xx:3,@aa:8 | 2 | | | 1 | | |
| | BTST #xx:3,@aa:16 | 3 | | | 1 | | |
| | BTST #xx:3,@aa:32 | 4 | | | 1 | | |
| | BTST Rn,Rd | 1 | | | | | |
| | BTST Rn,@ERd | 2 | | | 1 | | |
| | BTST Rn,@aa:8 | 2 | | | 1 | | |
| | BTST Rn,@aa:16 | 3 | | | 1 | | |
| | BTST Rn,@aa:32 | 4 | | | 1 | | |
| BXOR | BXOR #xx:3,Rd | 1 | | | | | |
| | BXOR #xx:3,@ERd | 2 | | | 1 | | |
| | BXOR #xx:3,@aa:8 | 2 | | | 1 | | |
| | BXOR #xx:3,@aa:16 | 3 | | | 1 | | |
| | BXOR #xx:3,@aa:32 | 4 | | | 1 | | |
| CLRMAC | CLRMAC | Cannot be used in the H8S/2345 Series | | | | | |
| CMP | CMP.B #xx:8,Rd | 1 | | | | | |
| | CMP.B Rs,Rd | 1 | | | | | |
| | CMP.W #xx:16,Rd | 2 | | | | | |
| | CMP.W Rs,Rd | 1 | | | | | |
| | CMP.L #xx:32,ERd | 3 | | | | | |
| | CMP.L ERs,ERd | 1 | | | | | |
| DAA | DAA Rd | 1 | | | | | |
| DAS | DAS Rd | 1 | | | | | |
| DEC | DEC.B Rd | 1 | | | | | |
| | DEC.W #1/2,Rd | 1 | | | | | |
| | DEC.L #1/2,ERd | 1 | | | | | |
| DIVXS | DIVXS.B Rs,Rd | 2 | | | | | 11 |
| | DIVXS.W Rs,ERd | 2 | | | | | 19 |
| DIVXU | DIVXU.B Rs,Rd | 1 | | | | | 11 |
| | DIVXU.W Rs,ERd | 1 | | | | | 19 |

**Table A.5 Number of Cycles in Instruction Execution (cont)**

| Instruction | Mnemonic | Instruction Fetch | Branch Address Read | Stack Operation | Byte Data Access | Word Data Access | Internal Operation |
|---|---|---|---|---|---|---|---|
| | | I | J | K | L | M | N |
| EEPMOV | EEPMOV.B | 2 | | | 2n+2*[2] | | |
| | EEPMOV.W | 2 | | | 2n+2*[2] | | |
| EXTS | EXTS.W Rd | 1 | | | | | |
| | EXTS.L ERd | 1 | | | | | |
| EXTU | EXTU.W Rd | 1 | | | | | |
| | EXTU.L ERd | 1 | | | | | |
| INC | INC.B Rd | 1 | | | | | |
| | INC.W #1/2,Rd | 1 | | | | | |
| | INC.L #1/2,ERd | 1 | | | | | |
| JMP | JMP @ERn | 2 | | | | | |
| | JMP @aa:24 | 2 | | | | | 1 |
| | JMP @@aa:8 | 2 | 2 | | | | 1 |
| JSR | JSR @ERn | 2 | | 2 | | | |
| | JSR @aa:24 | 2 | | 2 | | | 1 |
| | JSR @@aa:8 | 2 | 2 | 2 | | | |
| LDC | LDC #xx:8,CCR | 1 | | | | | |
| | LDC #xx:8,EXR | 2 | | | | | |
| | LDC Rs,CCR | 1 | | | | | |
| | LDC Rs,EXR | 1 | | | | | |
| | LDC @ERs,CCR | 2 | | | | 1 | |
| | LDC @ERs,EXR | 2 | | | | 1 | |
| | LDC @(d:16,ERs),CCR | 3 | | | | 1 | |
| | LDC @(d:16,ERs),EXR | 3 | | | | 1 | |
| | LDC @(d:32,ERs),CCR | 5 | | | | 1 | |
| | LDC @(d:32,ERs),EXR | 5 | | | | 1 | |
| | LDC @ERs+,CCR | 2 | | | | 1 | 1 |
| | LDC @ERs+,EXR | 2 | | | | 1 | 1 |
| | LDC @aa:16,CCR | 3 | | | | 1 | |
| | LDC @aa:16,EXR | 3 | | | | 1 | |
| | LDC @aa:32,CCR | 4 | | | | 1 | |
| | LDC @aa:32,EXR | 4 | | | | 1 | |

**Table A.5 Number of Cycles in Instruction Execution (cont)**

| Instruction | Mnemonic | Instruction Fetch | Branch Address Read | Stack Operation | Byte Data Access | Word Data Access | Internal Operation |
|---|---|---|---|---|---|---|---|
| | | I | J | K | L | M | N |
| LDM | LDM.L @SP+, (ERn-ERn+1) | 2 | | 4 | | | 1 |
| | LDM.L @SP+, (ERn-ERn+2) | 2 | | 6 | | | 1 |
| | LDM.L @SP+, (ERn-ERn+3) | 2 | | 8 | | | 1 |
| LDMAC | LDMAC ERs,MACH | Cannot be used in the H8S/2345 Series | | | | | |
| | LDMAC ERs,MACL | | | | | | |
| MAC | MAC @ERn+,@ERm+ | Cannot be used in the H8S/2345 Series | | | | | |
| MOV | MOV.B #xx:8,Rd | 1 | | | | | |
| | MOV.B Rs,Rd | 1 | | | | | |
| | MOV.B @ERs,Rd | 1 | | | 1 | | |
| | MOV.B @(d:16,ERs),Rd | 2 | | | 1 | | |
| | MOV.B @(d:32,ERs),Rd | 4 | | | 1 | | |
| | MOV.B @ERs+,Rd | 1 | | | 1 | | 1 |
| | MOV.B @aa:8,Rd | 1 | | | 1 | | |
| | MOV.B @aa:16,Rd | 2 | | | 1 | | |
| | MOV.B @aa:32,Rd | 3 | | | 1 | | |
| | MOV.B Rs,@ERd | 1 | | | 1 | | |
| | MOV.B Rs,@(d:16,ERd) | 2 | | | 1 | | |
| | MOV.B Rs,@(d:32,ERd) | 4 | | | 1 | | |
| | MOV.B Rs,@-ERd | 1 | | | 1 | | 1 |
| | MOV.B Rs,@aa:8 | 1 | | | 1 | | |
| | MOV.B Rs,@aa:16 | 2 | | | 1 | | |
| | MOV.B Rs,@aa:32 | 3 | | | 1 | | |
| | MOV.W #xx:16,Rd | 2 | | | | | |
| | MOV.W Rs,Rd | 1 | | | | | |
| | MOV.W @ERs,Rd | 1 | | | | 1 | |
| | MOV.W @(d:16,ERs),Rd | 2 | | | | 1 | |
| | MOV.W @(d:32,ERs),Rd | 4 | | | | 1 | |
| | MOV.W @ERs+,Rd | 1 | | | | 1 | 1 |
| | MOV.W @aa:16,Rd | 2 | | | | 1 | |
| | MOV.W @aa:32,Rd | 3 | | | | 1 | |
| | MOV.W Rs,@ERd | 1 | | | | 1 | |

**Table A.5 Number of Cycles in Instruction Execution (cont)**

| Instruction | Mnemonic | Instruction Fetch | Branch Address Read | Stack Operation | Byte Data Access | Word Data Access | Internal Operation |
|---|---|---|---|---|---|---|---|
| | | I | J | K | L | M | N |
| MOV | MOV.W Rs,@(d:16,ERd) | 2 | | | | 1 | |
| | MOV.W Rs,@(d:32,ERd) | 4 | | | | 1 | |
| | MOV.W Rs,@-ERd | 1 | | | | 1 | 1 |
| | MOV.W Rs,@aa:16 | 2 | | | | 1 | |
| | MOV.W Rs,@aa:32 | 3 | | | | 1 | |
| | MOV.L #xx:32,ERd | 3 | | | | | |
| | MOV.L ERs,ERd | 1 | | | | | |
| | MOV.L @ERs,ERd | 2 | | | | 2 | |
| | MOV.L @(d:16,ERs),ERd | 3 | | | | 2 | |
| | MOV.L @(d:32,ERs),ERd | 5 | | | | 2 | |
| | MOV.L @ERs+,ERd | 2 | | | | 2 | 1 |
| | MOV.L @aa:16,ERd | 3 | | | | 2 | |
| | MOV.L @aa:32,ERd | 4 | | | | 2 | |
| | MOV.L ERs,@ERd | 2 | | | | 2 | |
| | MOV.L ERs,@(d:16,ERd) | 3 | | | | 2 | |
| | MOV.L ERs,@(d:32,ERd) | 5 | | | | 2 | |
| | MOV.L ERs,@-ERd | 2 | | | | 2 | 1 |
| | MOV.L ERs,@aa:16 | 3 | | | | 2 | |
| | MOV.L ERs,@aa:32 | 4 | | | | 2 | |
| MOVFPE | MOVFPE @:aa:16,Rd | Can not be used in the H8S/2345 Series | | | | | |
| MOVTPE | MOVTPE Rs,@:aa:16 | | | | | | |
| MULXS | MULXS.B Rs,Rd | 2 | | | | | 11 |
| | MULXS.W Rs,ERd | 2 | | | | | 19 |
| MULXU | MULXU.B Rs,Rd | 1 | | | | | 11 |
| | MULXU.W Rs,ERd | 1 | | | | | 19 |
| NEG | NEG.B Rd | 1 | | | | | |
| | NEG.W Rd | 1 | | | | | |
| | NEG.L ERd | 1 | | | | | |
| NOP | NOP | 1 | | | | | |
| NOT | NOT.B Rd | 1 | | | | | |
| | NOT.W Rd | 1 | | | | | |
| | NOT.L ERd | 1 | | | | | |

**Table A.5 Number of Cycles in Instruction Execution (cont)**

| Instruction | Mnemonic | Instruction Fetch | Branch Address Read | Stack Operation | Byte Data Access | Word Data Access | Internal Operation |
|---|---|---|---|---|---|---|---|
| | | I | J | K | L | M | N |
| OR | OR.B #xx:8,Rd | 1 | | | | | |
| | OR.B Rs,Rd | 1 | | | | | |
| | OR.W #xx:16,Rd | 2 | | | | | |
| | OR.W Rs,Rd | 1 | | | | | |
| | OR.L #xx:32,ERd | 3 | | | | | |
| | OR.L ERs,ERd | 2 | | | | | |
| ORC | ORC #xx:8,CCR | 1 | | | | | |
| | ORC #xx:8,EXR | 2 | | | | | |
| POP | POP.W Rn | 1 | | | | 1 | 1 |
| | POP.L ERn | 2 | | | | 2 | 1 |
| PUSH | PUSH.W Rn | 1 | | | | 1 | 1 |
| | PUSH.L ERn | 2 | | | | 2 | 1 |
| ROTL | ROTL.B Rd | 1 | | | | | |
| | ROTL.B #2,Rd | 1 | | | | | |
| | ROTL.W Rd | 1 | | | | | |
| | ROTL.W #2,Rd | 1 | | | | | |
| | ROTL.L ERd | 1 | | | | | |
| | ROTL.L #2,ERd | 1 | | | | | |
| ROTR | ROTR.B Rd | 1 | | | | | |
| | ROTR.B #2,Rd | 1 | | | | | |
| | ROTR.W Rd | 1 | | | | | |
| | ROTR.W #2,Rd | 1 | | | | | |
| | ROTR.L ERd | 1 | | | | | |
| | ROTR.L #2,ERd | 1 | | | | | |
| ROTXL | ROTXL.B Rd | 1 | | | | | |
| | ROTXL.B #2,Rd | 1 | | | | | |
| | ROTXL.W Rd | 1 | | | | | |
| | ROTXL.W #2,Rd | 1 | | | | | |
| | ROTXL.L ERd | 1 | | | | | |
| | ROTXL.L #2,ERd | 1 | | | | | |

**Table A.5 Number of Cycles in Instruction Execution (cont)**

| Instruction | Mnemonic | Instruction Fetch | Branch Address Read | Stack Operation | Byte Data Access | Word Data Access | Internal Operation |
|---|---|---|---|---|---|---|---|
| | | I | J | K | L | M | N |
| ROTXR | ROTXR.B Rd | 1 | | | | | |
| | ROTXR.B #2,Rd | 1 | | | | | |
| | ROTXR.W Rd | 1 | | | | | |
| | ROTXR.W #2,Rd | 1 | | | | | |
| | ROTXR.L ERd | 1 | | | | | |
| | ROTXR.L #2,ERd | 1 | | | | | |
| RTE | RTE | 2 | | 2/3*[1] | | | 1 |
| RTS | RTS | 2 | | 2 | | | 1 |
| SHAL | SHAL.B Rd | 1 | | | | | |
| | SHAL.B #2,Rd | 1 | | | | | |
| | SHAL.W Rd | 1 | | | | | |
| | SHAL.W #2,Rd | 1 | | | | | |
| | SHAL.L ERd | 1 | | | | | |
| | SHAL.L #2,ERd | 1 | | | | | |
| SHAR | SHAR.B Rd | 1 | | | | | |
| | SHAR.B #2,Rd | 1 | | | | | |
| | SHAR.W Rd | 1 | | | | | |
| | SHAR.W #2,Rd | 1 | | | | | |
| | SHAR.L ERd | 1 | | | | | |
| | SHAR.L #2,ERd | 1 | | | | | |
| SHLL | SHLL.B Rd | 1 | | | | | |
| | SHLL.B #2,Rd | 1 | | | | | |
| | SHLL.W Rd | 1 | | | | | |
| | SHLL.W #2,Rd | 1 | | | | | |
| | SHLL.L ERd | 1 | | | | | |
| | SHLL.L #2,ERd | 1 | | | | | |
| SHLR | SHLR.B Rd | 1 | | | | | |
| | SHLR.B #2,Rd | 1 | | | | | |
| | SHLR.W Rd | 1 | | | | | |
| | SHLR.W #2,Rd | 1 | | | | | |
| | SHLR.L ERd | 1 | | | | | |
| | SHLR.L #2,ERd | 1 | | | | | |
| SLEEP | SLEEP | 1 | | | | | 1 |

**Table A.5 Number of Cycles in Instruction Execution (cont)**

| Instruction | Mnemonic | Instruction Fetch | Branch Address Read | Stack Operation | Byte Data Access | Word Data Access | Internal Operation |
|---|---|---|---|---|---|---|---|
| | | I | J | K | L | M | N |
| STC | STC.B CCR,Rd | 1 | | | | | |
| | STC.B EXR,Rd | 1 | | | | | |
| | STC.W CCR,@ERd | 2 | | | | 1 | |
| | STC.W EXR,@ERd | 2 | | | | 1 | |
| | STC.W CCR,@(d:16,ERd) | 3 | | | | 1 | |
| | STC.W EXR,@(d:16,ERd) | 3 | | | | 1 | |
| | STC.W CCR,@(d:32,ERd) | 5 | | | | 1 | |
| | STC.W EXR,@(d:32,ERd) | 5 | | | | 1 | |
| | STC.W CCR,@-ERd | 2 | | | | 1 | 1 |
| | STC.W EXR,@-ERd | 2 | | | | 1 | 1 |
| | STC.W CCR,@aa:16 | 3 | | | | 1 | |
| | STC.W EXR,@aa:16 | 3 | | | | 1 | |
| | STC.W CCR,@aa:32 | 4 | | | | 1 | |
| | STC.W EXR,@aa:32 | 4 | | | | 1 | |
| STM | STM.L (ERn-ERn+1), @-SP | 2 | | 4 | | | 1 |
| | STM.L (ERn-ERn+2), @-SP | 2 | | 6 | | | 1 |
| | STM.L (ERn-ERn+3), @-SP | 2 | | 8 | | | 1 |
| STMAC | STMAC MACH,ERd | Cannot be used in the H8S/2345 Series | | | | | |
| | STMAC MACL,ERd | | | | | | |
| SUB | SUB.B Rs,Rd | 1 | | | | | |
| | SUB.W #xx:16,Rd | 2 | | | | | |
| | SUB.W Rs,Rd | 1 | | | | | |
| | SUB.L #xx:32,ERd | 3 | | | | | |
| | SUB.L ERs,ERd | 1 | | | | | |
| SUBS | SUBS #1/2/4,ERd | 1 | | | | | |
| SUBX | SUBX #xx:8,Rd | 1 | | | | | |
| | SUBX Rs,Rd | 1 | | | | | |
| TAS | TAS @ERd | 2 | | | 2 | | |
| TRAPA | TRAPA #x:2 Advanced | 2 | 2 | 2/3*[1] | | | 2 |

**Table A.5 Number of Cycles in Instruction Execution (cont)**

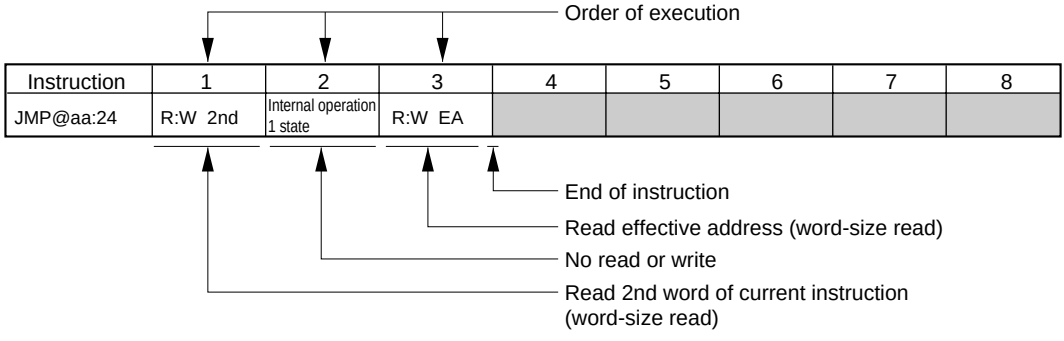
| Instruction | Mnemonic | Instruction Fetch | Branch Address Read | Stack Operation | Byte Data Access | Word Data Access | Internal Operation |
|---|---|---|---|---|---|---|---|
| | | I | J | K | L | M | N |
| XOR | XOR.B #xx:8,Rd | 1 | | | | | |
| | XOR.B Rs,Rd | 1 | | | | | |
| | XOR.W #xx:16,Rd | 2 | | | | | |
| | XOR.W Rs,Rd | 1 | | | | | |
| | XOR.L #xx:32,ERd | 3 | | | | | |
| | XOR.L ERs,ERd | 2 | | | | | |
| XORC | XORC #xx:8,CCR | 1 | | | | | |
| | XORC #xx:8,EXR | 2 | | | | | |

Notes: 1. 2 when EXR is invalid, 3 when EXR is valid.
2. When n bytes of data are transferred.

## A.5 Bus States During Instruction Execution

Table A.6 indicates the types of cycles that occur during instruction execution by the CPU. See table A.4 for the number of states per cycle.

**How to Read the Table:**




**Legend**

| | |
|---|---|
| R:B | Byte-size read |
| R:W | Word-size read |
| W:B | Byte-size write |
| W:W | Word-size write |
| :M | Transfer of the bus is not performed immediately after this cycle |
| 2nd | Address of 2nd word (3rd and 4th bytes) |
| 3rd | Address of 3rd word (5th and 6th bytes) |
| 4th | Address of 4th word (7th and 8th bytes) |
| 5th | Address of 5th word (9th and 10th bytes) |
| NEXT | Address of next instruction |
| EA | Effective address |
| VEC | Vector address |

Figure A.1 shows timing waveforms for the address bus and the $\overline{RD}$, $\overline{HWR}$, and $\overline{LWR}$ signals during execution of the above instruction with an 8-bit bus, using three-state access with no wait states.

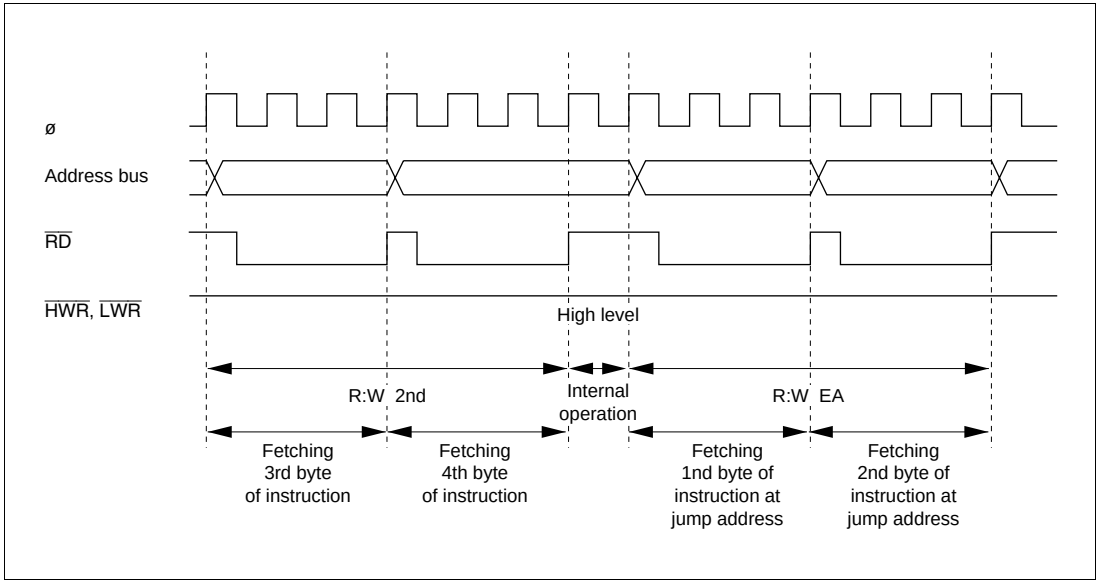



**Figure A.1 Address Bus, $\overline{RD}$, $\overline{HWR}$, and $\overline{LWR}$ Timing (8-Bit Bus, Three-State Access, No Wait States)**

**Table A.6 Instruction Execution Cycles**

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| ADD.B #xx:8,Rd | R:W NEXT | | | | | | | | |
| ADD.B Rs,Rd | R:W NEXT | | | | | | | | |
| ADD.W #xx:16,Rd | R:W 2nd | R:W NEXT | | | | | | | |
| ADD.W Rs,Rd | R:W NEXT | | | | | | | | |
| ADD.L #xx:32,ERd | R:W 2nd | R:W 3rd | R:W NEXT | | | | | | |
| ADD.L ERs,ERd | R:W NEXT | | | | | | | | |
| ADDS #1/2/4,ERd | R:W NEXT | | | | | | | | |
| ADDX #xx:8,Rd | R:W NEXT | | | | | | | | |
| ADDX Rs,Rd | R:W NEXT | | | | | | | | |
| AND.B #xx:8,Rd | R:W NEXT | | | | | | | | |
| AND.B Rs,Rd | R:W NEXT | | | | | | | | |
| AND.W #xx:16,Rd | R:W 2nd | R:W NEXT | | | | | | | |
| AND.W Rs,Rd | R:W NEXT | | | | | | | | |
| AND.L #xx:32,ERd | R:W 2nd | R:W 3rd | R:W NEXT | | | | | | |
| AND.L ERs,ERd | R:W 2nd | R:W NEXT | | | | | | | |
| ANDC #xx:8,CCR | R:W NEXT | | | | | | | | |
| ANDC #xx:8,EXR | R:W 2nd | R:W NEXT | | | | | | | |
| BAND #xx:3,Rd | R:W NEXT | | | | | | | | |
| BAND #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BAND #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BAND #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BAND #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BRA d:8 (BT d:8) | R:W NEXT | R:W EA | | | | | | | |
| BRN d:8 (BF d:8) | R:W NEXT | R:W EA | | | | | | | |
| BHI d:8 | R:W NEXT | R:W EA | | | | | | | |
| BLS d:8 | R:W NEXT | R:W EA | | | | | | | |
| BCC d:8 (BHS d:8) | R:W NEXT | R:W EA | | | | | | | |
| BCS d:8 (BLO d:8) | R:W NEXT | R:W EA | | | | | | | |
| BNE d:8 | R:W NEXT | R:W EA | | | | | | | |
| BEQ d:8 | R:W NEXT | R:W EA | | | | | | | |
| BVC d:8 | R:W NEXT | R:W EA | | | | | | | |
| BVS d:8 | R:W NEXT | R:W EA | | | | | | | |
| BPL d:8 | R:W NEXT | R:W EA | | | | | | | |
| BMI d:8 | R:W NEXT | R:W EA | | | | | | | |
| BGE d:8 | R:W NEXT | R:W EA | | | | | | | |
| BLT d:8 | R:W NEXT | R:W EA | | | | | | | |
| BGT d:8 | R:W NEXT | R:W EA | | | | | | | |

**Table A.6 Instruction Execution Cycles (cont)**

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| BLE d:8 | R:W NEXT | R:W EA | | | | | | | |
| BRA d:16 (BT d:16) | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BRN d:16 (BF d:16) | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BHI d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BLS d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BCC d:16 (BHS d:16) | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BCS d:16 (BLO d:16) | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BNE d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BEQ d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BVC d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BVS d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BPL d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BMI d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BGE d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BLT d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BGT d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BLE d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| BCLR #xx:3,Rd | R:W NEXT | | | | | | | | |
| BCLR #xx:3,@ERd | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BCLR #xx:3,@aa:8 | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BCLR #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |

**Table A.6 Instruction Execution Cycles (cont)**

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| BCLR #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BCLR Rn,Rd | R:W NEXT | | | | | | | | |
| BCLR Rn,@ERd | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BCLR Rn,@aa:8 | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BCLR Rn,@aa:16 | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |
| BCLR Rn,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BIAND #xx:3,Rd | R:W NEXT | | | | | | | | |
| BIAND #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BIAND #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BIAND #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BIAND #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BILD #xx:3,Rd | R:W NEXT | | | | | | | | |
| BILD #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BILD #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BILD #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BILD #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BIOR #xx:3,Rd | R:W NEXT | | | | | | | | |
| BIOR #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BIOR #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BIOR #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BIOR #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BIST #xx:3,Rd | R:W NEXT | | | | | | | | |
| BIST #xx:3,@ERd | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BIST #xx:3,@aa:8 | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BIST #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |
| BIST #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BIXOR #xx:3,Rd | R:W NEXT | | | | | | | | |
| BIXOR #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BIXOR #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BIXOR #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BIXOR #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BLD #xx:3,Rd | R:W NEXT | | | | | | | | |
| BLD #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BLD #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BLD #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BLD #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BNOT #xx:3,Rd | R:W NEXT | | | | | | | | |

**Table A.6 Instruction Execution Cycles (cont)**

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| BNOT #xx:3,@ERd | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BNOT #xx:3,@aa:8 | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BNOT #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |
| BNOT #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BNOT Rn,Rd | R:W NEXT | | | | | | | | |
| BNOT Rn,@ERd | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BNOT Rn,@aa:8 | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BNOT Rn,@aa:16 | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |
| BNOT Rn,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BOR #xx:3,Rd | R:W NEXT | | | | | | | | |
| BOR #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BOR #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BOR #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BOR #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BSET #xx:3,Rd | R:W NEXT | | | | | | | | |
| BSET #xx:3,@ERd | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BSET #xx:3,@aa:8 | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BSET #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |
| BSET #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BSET Rn,Rd | R:W NEXT | | | | | | | | |
| BSET Rn,@ERd | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BSET Rn,@aa:8 | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BSET Rn,@aa:16 | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |
| BSET Rn,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BSR d:8 | R:W NEXT | R:W EA | W:W:M stack (H) | W:W stack (L) | | | | | |
| BSR d:16 | R:W 2nd | Internal operation, 1 state | R:W EA | W:W:M stack (H) | W:W stack (L) | | | | |
| BST #xx:3,Rd | R:W NEXT | | | | | | | | |
| BST #xx:3,@ERd | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BST #xx:3,@aa:8 | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BST #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |
| BST #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BTST #xx:3,Rd | R:W NEXT | | | | | | | | |
| BTST #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |

## Table A.6 Instruction Execution Cycles (cont)

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| BTST #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BTST #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BTST #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BTST Rn,Rd | R:W NEXT | | | | | | | | |
| BTST Rn,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BTST Rn,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BTST Rn,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BTST Rn,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BXOR #xx:3,Rd | R:W NEXT | | | | | | | | |
| BXOR #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BXOR #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BXOR #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BXOR #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| CLRMAC | Cannot be used in the H8S/2345 Series | | | | | | | | |
| CMP.B #xx:8,Rd | R:W NEXT | | | | | | | | |
| CMP.B Rs,Rd | R:W NEXT | | | | | | | | |
| CMP.W #xx:16,Rd | R:W 2nd | R:W NEXT | | | | | | | |
| CMP.W Rs,Rd | R:W NEXT | | | | | | | | |
| CMP.L #xx:32,ERd | R:W 2nd | R:W 3rd | R:W NEXT | | | | | | |
| CMP.L ERs,ERd | R:W NEXT | | | | | | | | |
| DAA Rd | R:W NEXT | | | | | | | | |
| DAS Rd | R:W NEXT | | | | | | | | |
| DEC.B Rd | R:W NEXT | | | | | | | | |
| DEC.W #1/2,Rd | R:W NEXT | | | | | | | | |
| DEC.L #1/2,ERd | R:W NEXT | | | | | | | | |
| DIVXS.B Rs,Rd | R:W 2nd | R:W NEXT | Internal operation, 11 states | | | | | | |
| DIVXS.W Rs,ERd | R:W 2nd | R:W NEXT | Internal operation, 19 states | | | | | | |
| DIVXU.B Rs,Rd | R:W NEXT | Internal operation, 11 states | | | | | | | |
| DIVXU.W Rs,ERd | R:W NEXT | Internal operation, 19 states | | | | | | | |
| EEPMOV.B | R:W 2nd | R:B EAs*1 | R:B EAd*1 | R:B EAs*2 | W:B EAd*2 | R:W NEXT | | | |
| EEPMOV.W | R:W 2nd | R:B EAs*1 | R:B EAd*1 | R:B EAs*2 | W:B EAd*2 | R:W NEXT | | | |
| EXTS.W Rd | R:W NEXT | | | ← Repeated n times*2 → | | | | | |
| EXTS.L ERd | R:W NEXT | | | | | | | | |
| EXTU.W Rd | R:W NEXT | | | | | | | | |
| EXTU.L ERd | R:W NEXT | | | | | | | | |
| INC.B Rd | R:W NEXT | | | | | | | | |

**Table A.6 Instruction Execution Cycles (cont)**

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| INC.W #1/2,Rd | R:W NEXT | | | | | | | | |
| INC.L #1/2,ERd | R:W NEXT | | | | | | | | |
| JMP @ERn | R:W NEXT | R:W EA | | | | | | | |
| JMP @aa:24 | R:W 2nd | Internal operation, 1 state | R:W EA | | | | | | |
| JMP @@aa:8 | | | | | | | | | |
| | R:W NEXT | R:W:M aa:8 | R:W aa:8 | Internal operation, 1 state | R:W EA | | | | |
| JSR @ERn | | | | | | | | | |
| | R:W NEXT | R:W EA | W:W:M stack (H) | W:W stack (L) | | | | | |
| JSR @aa:24 | | | | | | | | | |
| | R:W 2nd | Internal operation, 1 state | R:W EA | W:W:M stack (H) | W:W stack (L) | | | | |
| JSR @@aa:8 | | | | | | | | | |
| | R:W NEXT | R:W:M aa:8 | R:W aa:8 | W:W:M stack (H) | W:W stack (L) | R:W EA | | | |
| LDC #xx:8,CCR | R:W NEXT | | | | | | | | |
| LDC #xx:8,EXR | R:W 2nd | R:W NEXT | | | | | | | |
| LDC Rs,CCR | R:W NEXT | | | | | | | | |
| LDC Rs,EXR | R:W NEXT | | | | | | | | |
| LDC @ERs,CCR | R:W 2nd | R:W NEXT | R:W EA | | | | | | |
| LDC @ERs,EXR | R:W 2nd | R:W NEXT | R:W EA | | | | | | |
| LDC @(d:16,ERs),CCR | R:W 2nd | R:W 3rd | R:W NEXT | R:W EA | | | | | |
| LDC @(d:16,ERs),EXR | R:W 2nd | R:W 3rd | R:W NEXT | R:W EA | | | | | |
| LDC @(d:32,ERs),CCR | R:W 2nd | R:W 3rd | R:W 4th | R:W 5th | R:W NEXT | R:W EA | | | |
| LDC @(d:32,ERs),EXR | R:W 2nd | R:W 3rd | R:W 4th | R:W 5th | R:W NEXT | R:W EA | | | |
| LDC @ERs+,CCR | R:W 2nd | R:W NEXT | Internal operation, 1 state | R:W EA | | | | | |
| LDC @ERs+,EXR | R:W 2nd | R:W NEXT | Internal operation, 1 state | R:W EA | | | | | |
| LDC @aa:16,CCR | R:W 2nd | R:W 3rd | R:W NEXT | R:W EA | | | | | |
| LDC @aa:16,EXR | R:W 2nd | R:W 3rd | R:W NEXT | R:W EA | | | | | |
| LDC @aa:32,CCR | R:W 2nd | R:W 3rd | R:W 4th | R:W NEXT | R:W EA | | | | |
| LDC @aa:32,EXR | R:W 2nd | R:W 3rd | R:W 4th | R:W NEXT | R:W EA | | | | |
| LDM.L @SP+, (ERn–ERn+1) | R:W 2nd | R:W:M NEXT | Internal operation, 1 state | R:W:M stack (H)*3 | R:W stack (L)*3 | | | | |

## Table A.6 Instruction Execution Cycles (cont)

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| LDM.L @SP+,(ERn–ERn+2) | R:W 2nd | R:W NEXT | Internal operation, 1 state | R:W:M stack (H)*3 | R:W stack (L)*3 | | | | |
| LDM.L @SP+,(ERn–ERn+3) | R:W 2nd | R:W NEXT | Internal operation, 1 state | R:W:M stack (H)*3 | R:W stack (L)*3 | | | | |
| LDMAC ERs,MACH | Cannot be used in the H8S/2345 Series | | | | | | | | |
| LDMAC ERs,MACL | | | | | | | | | |
| MAC @ERn+,@ERm+ | | | | | | | | | |
| MOV.B #xx:8,Rd | R:W NEXT | | | | | | | | |
| MOV.B Rs,Rd | R:W NEXT | | | | | | | | |
| MOV.B @ERs,Rd | R:W NEXT | R:B EA | | | | | | | |
| MOV.B @(d:16,ERs),Rd | R:W 2nd | R:W NEXT | R:B EA | | | | | | |
| MOV.B @(d:32,ERs),Rd | R:W 2nd | R:W 3rd | R:W 4th | R:W NEXT | R:B EA | | | | |
| MOV.B @ERs+,Rd | R:W NEXT | Internal operation, 1 state | R:B EA | | | | | | |
| MOV.B @aa:8,Rd | R:W NEXT | R:B EA | | | | | | | |
| MOV.B @aa:16,Rd | R:W 2nd | R:W NEXT | R:B EA | | | | | | |
| MOV.B @aa:32,Rd | R:W 2nd | R:W 3rd | R:W NEXT | R:B EA | | | | | |
| MOV.B Rs,@ERd | R:W NEXT | W:B EA | | | | | | | |
| MOV.B Rs,@(d:16,ERd) | R:W 2nd | R:W NEXT | W:B EA | | | | | | |
| MOV.B Rs,@(d:32,ERd) | R:W 2nd | R:W 3rd | R:W 4th | R:W NEXT | W:B EA | | | | |
| MOV.B Rs,@–ERd | R:W NEXT | Internal operation, 1 state | W:B EA | | | | | | |
| MOV.B Rs,@aa:8 | R:W NEXT | W:B EA | | | | | | | |
| MOV.B Rs,@aa:16 | R:W 2nd | R:W NEXT | W:B EA | | | | | | |
| MOV.B Rs,@aa:32 | R:W 2nd | R:W 3rd | R:W NEXT | W:B EA | | | | | |
| MOV.W #xx:16,Rd | R:W 2nd | R:W NEXT | | | | | | | |
| MOV.W Rs,Rd | R:W NEXT | | | | | | | | |
| MOV.W @ERs,Rd | R:W NEXT | R:W EA | | | | | | | |
| MOV.W @(d:16,ERs),Rd | R:W 2nd | R:W NEXT | R:W EA | | | | | | |
| MOV.W @(d:32,ERs),Rd | R:W 2nd | R:W 3rd | R:W 4th | R:W NEXT | R:W EA | | | | |
| MOV.W @ERs+, Rd | R:W NEXT | Internal operation, 1 state | R:W EA | | | | | | |
| MOV.W @aa:16,Rd | R:W 2nd | R:W NEXT | R:W EA | | | | | | |
| MOV.W @aa:32,Rd | R:W 2nd | R:W 3rd | R:W NEXT | R:B EA | | | | | |
| MOV.W Rs,@ERd | R:W NEXT | W:W EA | | | | | | | |

**Table A.6 Instruction Execution Cycles (cont)**

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| MOV.W Rs,@(d:16,ERd) | R:W 2nd | R:W NEXT | W:W EA | | | | | | |
| MOV.W Rs,@(d:32,ERd) | R:W 2nd | R:W 3rd | R:E 4th | R:W NEXT | W:W EA | | | | |
| MOV.W Rs,@–ERd | R:W NEXT | Internal operation, 1 state | W:W EA | | | | | | |
| MOV.W Rs,@aa:16 | R:W 2nd | R:W NEXT | W:W EA | | | | | | |
| MOV.W Rs,@aa:32 | R:W 2nd | R:W 3rd | R:W NEXT | W:W EA | | | | | |
| MOV.L #xx:32,ERd | R:W 2nd | R:W 3rd | R:W NEXT | | | | | | |
| MOV.L ERs,ERd | R:W NEXT | | | | | | | | |
| MOV.L @ERs,ERd | R:W 2nd | R:W:M NEXT | R:W:M EA | R:W EA+2 | | | | | |
| MOV.L @(d:16,ERs),ERd | R:W 2nd | R:W:M 3rd | R:W NEXT | R:W:M EA | R:W EA+2 | | | | |
| MOV.L @(d:32,ERs),ERd | R:W 2nd | R:W:M 3rd | R:W:M 4th | R:W 5th | R:W NEXT | R:W:M EA | R:W EA+2 | | |
| MOV.L @ERs+,ERd | R:W 2nd | R:W:M NEXT | Internal operation, 1 state | R:W:M EA | R:W EA+2 | | | | |
| MOV.L @aa:16,ERd | R:W 2nd | R:W:M 3rd | R:W NEXT | R:W:M EA | R:W EA+2 | | | | |
| MOV.L @aa:32,ERd | R:W 2nd | R:W:M 3rd | R:W 4th | R:W NEXT | R:W:M EA | R:W EA+2 | | | |
| MOV.L ERs,@ERd | R:W 2nd | R:W:M NEXT | W:W:M EA | W:W EA+2 | | | | | |
| MOV.L ERs,@(d:16,ERd) | R:W 2nd | R:W:M 3rd | R:W NEXT | W:W:M EA | W:W EA+2 | | | | |
| MOV.L ERs,@(d:32,ERd) | R:W 2nd | R:W:M 3rd | R:W:M 4th | R:W 5th | R:W NEXT | W:W:M EA | W:W EA+2 | | |
| MOV.L ERs,@–ERd | R:W 2nd | R:W:M NEXT | Internal operation, 1 state | W:W:M EA | W:W EA+2 | | | | |
| MOV.L ERs,@aa:16 | R:W 2nd | R:W:M 3rd | R:W NEXT | W:W:M EA | W:W EA+2 | | | | |
| MOV.L ERs,@aa:32 | R:W 2nd | R:W:M 3rd | R:W 4th | R:W NEXT | W:W:M EA | W:W EA+2 | | | |
| MOVFPE @aa:16,Rd | Cannot be used in the H8S/2345 Series | | | | | | | | |
| MOVTPE Rs,@aa:16 | | | | | | | | | |
| MULXS.B Rs,Rd | R:W 2nd | R:W NEXT | Internal operation, 11 states | | | | | | |
| MULXS.W Rs,ERd | R:W 2nd | R:W NEXT | Internal operation, 19 states | | | | | | |
| MULXU.B Rs,Rd | R:W NEXT | Internal operation, 11 states | | | | | | | |
| MULXU.W Rs,ERd | R:W NEXT | Internal operation, 19 states | | | | | | | |
| NEG.B Rd | R:W NEXT | | | | | | | | |
| NEG.W Rd | R:W NEXT | | | | | | | | |
| NEG.L ERd | R:W NEXT | | | | | | | | |
| NOP | R:W NEXT | | | | | | | | |
| NOT.B Rd | R:W NEXT | | | | | | | | |
| NOT.W Rd | R:W NEXT | | | | | | | | |
| NOT.L ERd | R:W NEXT | | | | | | | | |
| OR.B #xx:8,Rd | R:W NEXT | | | | | | | | |
| OR.B Rs,Rd | R:W NEXT | | | | | | | | |

**Table A.6 Instruction Execution Cycles (cont)**

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| OR.W #xx:16,Rd | R:W 2nd | R:W NEXT | | | | | | | |
| OR.W Rs,Rd | R:W NEXT | | | | | | | | |
| OR.L #xx:32,ERd | R:W 2nd | R:W 3rd | R:W NEXT | | | | | | |
| OR.L ERs,ERd | R:W 2nd | R:W NEXT | | | | | | | |
| ORC #xx:8,CCR | R:W NEXT | | | | | | | | |
| ORC #xx:8,EXR | R:W 2nd | R:W NEXT | | | | | | | |
| POP.W Rn | R:W NEXT | Internal operation, 1 state | R:W EA | | | | | | |
| POP.L ERn | R:W 2nd | R:W:M NEXT | Internal operation, 1 state | R:W:M EA | R:W EA+2 | | | | |
| PUSH.W Rn | R:W NEXT | Internal operation, 1 state | W:W EA | | | | | | |
| PUSH.L ERn | R:W 2nd | R:W:M NEXT | Internal operation, 1 state | W:W:M EA | W:W EA+2 | | | | |
| ROTL.B Rd | R:W NEXT | | | | | | | | |
| ROTL.B #2,Rd | R:W NEXT | | | | | | | | |
| ROTL.W Rd | R:W NEXT | | | | | | | | |
| ROTL.W #2,Rd | R:W NEXT | | | | | | | | |
| ROTL.L ERd | R:W NEXT | | | | | | | | |
| ROTL.L #2,ERd | R:W NEXT | | | | | | | | |
| ROTR.B Rd | R:W NEXT | | | | | | | | |
| ROTR.B #2,Rd | R:W NEXT | | | | | | | | |
| ROTR.W Rd | R:W NEXT | | | | | | | | |
| ROTR.W #2,Rd | R:W NEXT | | | | | | | | |
| ROTR.L ERd | R:W NEXT | | | | | | | | |
| ROTR.L #2,ERd | R:W NEXT | | | | | | | | |
| ROTXL.B Rd | R:W NEXT | | | | | | | | |
| ROTXL.B #2,Rd | R:W NEXT | | | | | | | | |
| ROTXL.W Rd | R:W NEXT | | | | | | | | |
| ROTXL.W #2,Rd | R:W NEXT | | | | | | | | |
| ROTXL.L ERd | R:W NEXT | | | | | | | | |
| ROTXL.L #2,ERd | R:W NEXT | | | | | | | | |
| ROTXR.B Rd | R:W NEXT | | | | | | | | |
| ROTXR.B #2,Rd | R:W NEXT | | | | | | | | |
| ROTXR.W Rd | R:W NEXT | | | | | | | | |
| ROTXR.W #2,Rd | R:W NEXT | | | | | | | | |
| ROTXR.L ERd | R:W NEXT | | | | | | | | |

**Table A.6 Instruction Execution Cycles (cont)**

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| ROTXR.L #2,ERd | R:W NEXT | | | | | | | | |
| RTE | R:W NEXT | R:W stack (EXR) | R:W stack (H) | R:W stack (L) | Internal operation, 1 state | R:W*4 | | | |
| RTS | R:W NEXT | R:W:M stack (H) | R:W stack (L) | Internal operation, 1 state | R:W*4 | | | | |
| SHAL.B Rd | R:W NEXT | | | | | | | | |
| SHAL.B #2,Rd | R:W NEXT | | | | | | | | |
| SHAL.W Rd | R:W NEXT | | | | | | | | |
| SHAL.W #2,Rd | R:W NEXT | | | | | | | | |
| SHAL.L ERd | R:W NEXT | | | | | | | | |
| SHAL.L #2,ERd | R:W NEXT | | | | | | | | |
| SHAR.B Rd | R:W NEXT | | | | | | | | |
| SHAR.B #2,Rd | R:W NEXT | | | | | | | | |
| SHAR.W Rd | R:W NEXT | | | | | | | | |
| SHAR.W #2,Rd | R:W NEXT | | | | | | | | |
| SHAR.L ERd | R:W NEXT | | | | | | | | |
| SHAR.L #2,ERd | R:W NEXT | | | | | | | | |
| SHLL.B Rd | R:W NEXT | | | | | | | | |
| SHLL.B #2,Rd | R:W NEXT | | | | | | | | |
| SHLL.W Rd | R:W NEXT | | | | | | | | |
| SHLL.W #2,Rd | R:W NEXT | | | | | | | | |
| SHLL.L ERd | R:W NEXT | | | | | | | | |
| SHLL.L #2,ERd | R:W NEXT | | | | | | | | |
| SHLR.B Rd | R:W NEXT | | | | | | | | |
| SHLR.B #2,Rd | R:W NEXT | | | | | | | | |
| SHLR.W Rd | R:W NEXT | | | | | | | | |
| SHLR.W #2,Rd | R:W NEXT | | | | | | | | |
| SHLR.L ERd | R:W NEXT | | | | | | | | |
| SHLR.L #2,ERd | R:W NEXT | | | | | | | | |
| SLEEP | R:W NEXT | Internal operation:M | | | | | | | |
| STC CCR,Rd | R:W NEXT | | | | | | | | |
| STC EXR,Rd | R:W NEXT | | | | | | | | |
| STC CCR,@ERd | R:W 2nd | R:W NEXT | W:W EA | | | | | | |
| STC EXR,@ERd | R:W 2nd | R:W NEXT | W:W EA | | | | | | |
| STC CCR,@(d:16,ERd) | R:W 2nd | R:W 3rd | R:W NEXT | W:W EA | | | | | |

## Table A.6 Instruction Execution Cycles (cont)

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| STC EXR,@(d:16,ERd) | R:W 2nd | R:W 3rd | R:W NEXT | W:W EA | | | | | |
| STC CCR,@(d:32,ERd) | R:W 2nd | R:W 3rd | R:W 4th | R:W 5th | R:W NEXT | W:W EA | | | |
| STC EXR,@(d:32,ERd) | R:W 2nd | R:W 3rd | R:W 4th | R:W 5th | R:W NEXT | W:W EA | | | |
| STC CCR,@–ERd | R:W 2nd | R:W NEXT | Internal operation, 1 state | W:W EA | | | | | |
| STC EXR,@–ERd | R:W 2nd | R:W NEXT | Internal operation, 1 state | W:W EA | | | | | |
| STC CCR,@aa:16 | R:W 2nd | R:W 3rd | R:W NEXT | W:W EA | | | | | |
| STC EXR,@aa:16 | R:W 2nd | R:W 3rd | R:W NEXT | W:W EA | | | | | |
| STC CCR,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:W NEXT | W:W EA | | | | |
| STC EXR,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:W NEXT | W:W EA | | | | |
| STM.L(ERn–ERn+1),@–SP | R:W 2nd | R:W:M NEXT | Internal operation, 1 state | W:W:M stack (H)*3 | W:W stack (L)*3 | | | | |
| STM.L(ERn–ERn+2),@–SP | R:W 2nd | R:W:M NEXT | Internal operation, 1 state | W:W:M stack (H)*3 | W:W stack (L)*3 | | | | |
| STM.L(ERn–ERn+3),@–SP | R:W 2nd | R:W:M NEXT | Internal operation, 1 state | W:W:M stack (H)*3 | W:W stack (L)*3 | | | | |
| STMAC MACH,ERd | Cannot be used in the H8S/2345 Series | | | | | | | | |
| STMAC MACL,ERd | | | | | | | | | |
| SUB.B Rs,Rd | R:W NEXT | | | | | | | | |
| SUB.W #xx:16,Rd | R:W 2nd | R:W NEXT | | | | | | | |
| SUB.W Rs,Rd | R:W NEXT | | | | | | | | |
| SUB.L #xx:32,ERd | R:W 2nd | R:W 3rd | R:W NEXT | | | | | | |
| SUB.L ERs,ERd | R:W NEXT | | | | | | | | |
| SUBS #1/2/4,ERd | R:W NEXT | | | | | | | | |
| SUBX #xx:8,Rd | R:W NEXT | | | | | | | | |
| SUBX Rs,Rd | R:W NEXT | | | | | | | | |
| TAS @ERd | R:W 2nd | R:W NEXT | R:B:M EA | W:B EA | | | | | |
| TRAPA #x:2 | R:W NEXT | Internal operation, 1 state | W:W stack (L) | W:W stack (H) | W:W stack (EXR) | R:W:M VEC | R:W VEC+2 | Internal operation, 1 state | R:W*7 |
| XOR.B #xx8,Rd | R:W NEXT | | | | | | | | |
| XOR.B Rs,Rd | R:W NEXT | | | | | | | | |
| XOR.W #xx:16,Rd | R:W 2nd | R:W NEXT | | | | | | | |
| XOR.W Rs,Rd | R:W NEXT | | | | | | | | |
| XOR.L #xx:32,ERd | R:W 2nd | R:W 3rd | R:W NEXT | | | | | | |

**Table A.6 Instruction Execution Cycles (cont)**

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| XOR.L ERs,ERd | R:W 2nd | R:W NEXT | | | | | | | |
| XORC #xx:8,CCR | R:W NEXT | | | | | | | | |
| XORC #xx:8,EXR | R:W 2nd | R:W NEXT | | | | | | | |
| Reset exception handling | R:W:M VEC | R:W VEC+2 | Internal operation, 1 state | R:W*5 | | | | | |
| Interrupt exception handling | R:W*6 | Internal operation, 1 state | W:W stack (L) | W:W stack (H) | W:W stack (EXR) | R:W:M VEC | R:W VEC+2 | Internal operation, 1 state | R:W*7 |

Notes:
1. EAs is the contents of ER5. EAd is the contents of ER6.
2. EAs is the contents of ER5. EAd is the contents of ER6. Both registers are incremented by 1 after execution of the instruction. n is the initial value of R4L or R4. If n = 0, these bus cycles are not executed.
3. Repeated two times to save or restore two registers, three times for three registers, or four times for four registers.
4. Start address after return.
5. Start address of the program.
6. Prefetch address, equal to two plus the PC value pushed onto the stack. In recovery from sleep mode or software standby mode the read operation is replaced by an internal operation.
7. Start address of the interrupt-handling routine.

## A.6 Condition Code Modification

This section indicates the effect of each CPU instruction on the condition code. The notation used in the table is defined below.

m = { 31 for longword operands; 15 for word operands; 7 for byte operands

| | |
|---|---|
| Si | The i-th bit of the source operand |
| Di | The i-th bit of the destination operand |
| Ri | The i-th bit of the result |
| Dn | The specified bit in the destination operand |
| — | Not affected |
| ↕ | Modified according to the result of the instruction (see definition) |
| 0 | Always cleared to 0 |
| 1 | Always set to 1 |
| * | Undetermined (no guaranteed value) |
| Z' | Z flag before instruction execution |
| C' | C flag before instruction execution |

**Table A.7 Condition Code Modification**

| Instruction | H | N | Z | V | C | Definition |
|---|---|---|---|---|---|---|
| ADD | ↕ | ↕ | ↕ | ↕ | ↕ | H = Sm–4 · Dm–4 + Dm–4 · $\overline{Rm–4}$ + Sm–4 · $\overline{Rm–4}$<br>N = Rm<br>Z = $\overline{Rm}$ · $\overline{Rm–1}$ · ····· · $\overline{R0}$<br>V = Sm · Dm · $\overline{Rm}$ + $\overline{Sm}$ · $\overline{Dm}$ · Rm<br>C = Sm · Dm + Dm · $\overline{Rm}$ + Sm · $\overline{Rm}$ |
| ADDS | — | — | — | — | — | |
| ADDX | ↕ | ↕ | ↕ | ↕ | ↕ | H = Sm–4 · Dm–4 + Dm–4 · $\overline{Rm–4}$ + Sm–4 · $\overline{Rm–4}$<br>N = Rm<br>Z = Z' · $\overline{Rm}$ · ····· · $\overline{R0}$<br>V = Sm · Dm · $\overline{Rm}$ + $\overline{Sm}$ · $\overline{Dm}$ · Rm<br>C = Sm · Dm + Dm · $\overline{Rm}$ + Sm · $\overline{Rm}$ |
| AND | — | ↕ | ↕ | 0 | — | N = Rm<br>Z = $\overline{Rm}$ · $\overline{Rm–1}$ · ····· · $\overline{R0}$ |
| ANDC | ↕ | ↕ | ↕ | ↕ | ↕ | Stores the corresponding bits of the result.<br>No flags change when the operand is EXR. |
| BAND | — | — | — | — | ↕ | C = C' · Dn |
| Bcc | — | — | — | — | — | |
| BCLR | — | — | — | — | — | |
| BIAND | — | — | — | — | ↕ | C = C' · $\overline{Dn}$ |
| BILD | — | — | — | — | ↕ | C = $\overline{Dn}$ |
| BIOR | — | — | — | — | ↕ | C = C' + $\overline{Dn}$ |
| BIST | — | — | — | — | — | |
| BIXOR | — | — | — | — | ↕ | C = C' · Dn + $\overline{C'}$ · $\overline{Dn}$ |
| BLD | — | — | — | — | ↕ | C = Dn |
| BNOT | — | — | — | — | — | |
| BOR | — | — | — | — | ↕ | C = C' + Dn |
| BSET | — | — | — | — | — | |
| BSR | — | — | — | — | — | |
| BST | — | — | — | — | — | |
| BTST | — | — | ↕ | — | — | Z = $\overline{Dn}$ |
| BXOR | — | — | — | — | ↕ | C = C' · $\overline{Dn}$ + $\overline{C'}$ · Dn |
| CLRMAC | | | | | | Cannot be used in the H8S/2345 Series |

**Table A.7 Condition Code Modification (cont)**

| Instruction | H | N | Z | V | C | Definition |
|---|---|---|---|---|---|---|
| CMP | ↕ | ↕ | ↕ | ↕ | ↕ | H = Sm–4 · $\overline{Dm–4}$ + $\overline{Dm–4}$ · Rm–4 + Sm–4 · Rm–4 |
| | | | | | | N = Rm |
| | | | | | | Z = $\overline{Rm}$ · $\overline{Rm–1}$ · ····· · $\overline{R0}$ |
| | | | | | | V = $\overline{Sm}$ · Dm · $\overline{Rm}$ + Sm · $\overline{Dm}$ · Rm |
| | | | | | | C = Sm · $\overline{Dm}$ + $\overline{Dm}$ · Rm + Sm · Rm |
| DAA | * | ↕ | ↕ | * | ↕ | N = Rm |
| | | | | | | Z = $\overline{Rm}$ · $\overline{Rm–1}$ · ····· · $\overline{R0}$ |
| | | | | | | C: decimal arithmetic carry |
| DAS | * | ↕ | ↕ | * | ↕ | N = Rm |
| | | | | | | Z = $\overline{Rm}$ · $\overline{Rm–1}$ · ····· · $\overline{R0}$ |
| | | | | | | C: decimal arithmetic borrow |
| DEC | — | ↕ | ↕ | ↕ | — | N = Rm |
| | | | | | | Z = $\overline{Rm}$ · $\overline{Rm–1}$ · ····· · $\overline{R0}$ |
| | | | | | | V = Dm · $\overline{Rm}$ |
| DIVXS | — | ↕ | ↕ | — | — | N = Sm · $\overline{Dm}$ + $\overline{Sm}$ · Dm |
| | | | | | | Z = $\overline{Sm}$ · $\overline{Sm–1}$ · ····· · $\overline{S0}$ |
| DIVXU | — | ↕ | ↕ | — | — | N = Sm |
| | | | | | | Z = $\overline{Sm}$ · $\overline{Sm–1}$ · ····· · $\overline{S0}$ |
| EEPMOV | — | — | — | — | — | |
| EXTS | — | ↕ | ↕ | 0 | — | N = Rm |
| | | | | | | Z = $\overline{Rm}$ · $\overline{Rm–1}$ · ····· · $\overline{R0}$ |
| EXTU | — | 0 | ↕ | 0 | — | Z = $\overline{Rm}$ · $\overline{Rm–1}$ · ····· · $\overline{R0}$ |
| INC | — | ↕ | ↕ | ↕ | — | N = Rm |
| | | | | | | Z = $\overline{Rm}$ · $\overline{Rm–1}$ · ····· · $\overline{R0}$ |
| | | | | | | V = $\overline{Dm}$ · Rm |
| JMP | — | — | — | — | — | |
| JSR | — | — | — | — | — | |
| LDC | ↕ | ↕ | ↕ | ↕ | ↕ | Stores the corresponding bits of the result. |
| | | | | | | No flags change when the operand is EXR. |
| LDM | — | — | — | — | — | |
| LDMAC | | | | | | Cannnot be used in the H8S/2345 Series |
| MAC | | | | | | |

**Table A.7 Condition Code Modification (cont)**

| Instruction | H | N | Z | V | C | Definition |
|---|---|---|---|---|---|---|
| MOV | — | ↕ | ↕ | 0 | — | N = Rm<br>Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$ |
| MOVFPE | | | | | | Can not be used in the H8S/2345 Series |
| MOVTPE | | | | | | |
| MULXS | — | ↕ | ↕ | — | — | N = R2m<br>Z = $\overline{R2m} \cdot \overline{R2m-1} \cdot \cdots\cdots \cdot \overline{R0}$ |
| MULXU | — | — | — | — | — | |
| NEG | ↕ | ↕ | ↕ | ↕ | ↕ | H = Dm–4 + Rm–4<br>N = Rm<br>Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$<br>V = Dm · Rm<br>C = Dm + Rm |
| NOP | — | — | — | — | — | |
| NOT | — | ↕ | ↕ | 0 | — | N = Rm<br>Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$ |
| OR | — | ↕ | ↕ | 0 | — | N = Rm<br>Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$ |
| ORC | ↕ | ↕ | ↕ | ↕ | ↕ | Stores the corresponding bits of the result.<br>No flags change when the operand is EXR. |
| POP | — | ↕ | ↕ | 0 | — | N = Rm<br>Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$ |
| PUSH | — | ↕ | ↕ | 0 | — | N = Rm<br>Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$ |
| ROTL | — | ↕ | ↕ | 0 | ↕ | N = Rm<br>Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$<br>C = Dm (1-bit shift) or C = Dm–1 (2-bit shift) |
| ROTR | — | ↕ | ↕ | 0 | ↕ | N = Rm<br>Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$<br>C = D0 (1-bit shift) or C = D1 (2-bit shift) |

**Table A.7 Condition Code Modification (cont)**

| Instruction | H | N | Z | V | C | Definition |
|---|---|---|---|---|---|---|
| ROTXL | — | ↕ | ↕ | 0 | ↕ | N = Rm |
| | | | | | | Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$ |
| | | | | | | C = Dm (1-bit shift) or C = Dm–1 (2-bit shift) |
| ROTXR | — | ↕ | ↕ | 0 | ↕ | N = Rm |
| | | | | | | Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$ |
| | | | | | | C = D0 (1-bit shift) or C = D1 (2-bit shift) |
| RTE | ↕ | ↕ | ↕ | ↕ | ↕ | Stores the corresponding bits of the result. |
| RTS | — | — | — | — | — | |
| SHAL | — | ↕ | ↕ | ↕ | ↕ | N = Rm |
| | | | | | | Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$ |
| | | | | | | V = $\overline{Dm \cdot Dm-1 + \overline{Dm} \cdot \overline{Dm-1}}$ (1-bit shift) |
| | | | | | | V = $\overline{Dm \cdot Dm-1 \cdot Dm-2 \cdot \overline{Dm} \cdot \overline{Dm-1} \cdot \overline{Dm-2}}$ (2-bit shift) |
| | | | | | | C = Dm (1-bit shift) or C = Dm–1 (2-bit shift) |
| SHAR | — | ↕ | ↕ | 0 | ↕ | N = Rm |
| | | | | | | Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$ |
| | | | | | | C = D0 (1-bit shift) or C = D1 (2-bit shift) |
| SHLL | — | ↕ | ↕ | 0 | ↕ | N = Rm |
| | | | | | | Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$ |
| | | | | | | C = Dm (1-bit shift) or C = Dm–1 (2-bit shift) |
| SHLR | — | 0 | ↕ | 0 | ↕ | N = Rm |
| | | | | | | Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \cdots\cdots \cdot \overline{R0}$ |
| | | | | | | C = D0 (1-bit shift) or C = D1 (2-bit shift) |
| SLEEP | — | — | — | — | — | |
| STC | — | — | — | — | — | |
| STM | — | — | — | — | — | |
| STMAC | | | | | | Cannot be used in the H8S/2345 Series |

**Table A.7 Condition Code Modification (cont)**

| Instruction | H | N | Z | V | C | Definition |
|---|---|---|---|---|---|---|
| SUB | ↕ | ↕ | ↕ | ↕ | ↕ | H = Sm–4 · $\overline{Dm–4}$ + $\overline{Dm–4}$ · Rm–4 + Sm–4 · Rm–4 |
| | | | | | | N = Rm |
| | | | | | | Z = $\overline{Rm}$ · $\overline{Rm–1}$ · ····· · $\overline{R0}$ |
| | | | | | | V = $\overline{Sm}$ · Dm · $\overline{Rm}$ + Sm · $\overline{Dm}$ · Rm |
| | | | | | | C = Sm · $\overline{Dm}$ + $\overline{Dm}$ · Rm + Sm · Rm |
| SUBS | — | — | — | — | — | |
| SUBX | ↕ | ↕ | ↕ | ↕ | ↕ | H = Sm–4 · $\overline{Dm–4}$ + $\overline{Dm–4}$ · Rm–4 + Sm–4 · Rm–4 |
| | | | | | | N = Rm |
| | | | | | | Z = Z' · $\overline{Rm}$ · ····· · $\overline{R0}$ |
| | | | | | | V = $\overline{Sm}$ · Dm · $\overline{Rm}$ + Sm · $\overline{Dm}$ · Rm |
| | | | | | | C = Sm · $\overline{Dm}$ + $\overline{Dm}$ · Rm + Sm · Rm |
| TAS | — | ↕ | ↕ | 0 | — | N = Dm |
| | | | | | | Z = $\overline{Dm}$ · $\overline{Dm–1}$ · ····· · $\overline{D0}$ |
| TRAPA | — | — | — | — | — | |
| XOR | — | ↕ | ↕ | 0 | — | N = Rm |
| | | | | | | Z = $\overline{Rm}$ · $\overline{Rm–1}$ · ····· · $\overline{R0}$ |
| XORC | ↕ | ↕ | ↕ | ↕ | ↕ | Stores the corresponding bits of the result. |
| | | | | | | No flags change when the operand is EXR. |

# Appendix B Internal I/O Register

## B.1 Addresses

| Address (low) | Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module Name | Data Bus Width |
|---|---|---|---|---|---|---|---|---|---|---|---|
| H'F800 | MRA | SM1 | SM0 | DM1 | DM0 | MD1 | MD0 | DTS | Sz | DTC | 16/32* bit |
| to H'FBFF | SAR | | | | | | | | | | |
| | MRB | CHNE | DISEL | — | — | — | — | — | — | | |
| | DAR | | | | | | | | | | |
| | CRA | | | | | | | | | | |
| | CRB | | | | | | | | | | |
| H'FE80 | TCR3 | CCLR2 | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 | TPU3 | 16 bit |
| H'FE81 | TMDR3 | — | — | BFB | BFA | MD3 | MD2 | MD1 | MD0 | | |
| H'FE82 | TIOR3H | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 | | |
| H'FE83 | TIOR3L | IOD3 | IOD2 | IOD1 | IOD0 | IOC3 | IOC2 | IOC1 | IOC0 | | |
| H'FE84 | TIER3 | TTGE | — | — | TCIEV | TGIED | TGIEC | TGIEB | TGIEA | | |
| H'FE85 | TSR3 | — | — | — | TCFV | TGFD | TGFC | TGFB | TGFA | | |
| H'FE86 | TCNT3 | | | | | | | | | | |
| H'FE87 | | | | | | | | | | | |
| H'FE88 | TGR3A | | | | | | | | | | |
| H'FE89 | | | | | | | | | | | |
| H'FE8A | TGR3B | | | | | | | | | | |
| H'FE8B | | | | | | | | | | | |
| H'FE8C | TGR3C | | | | | | | | | | |
| H'FE8D | | | | | | | | | | | |
| H'FE8E | TGR3D | | | | | | | | | | |
| H'FE8F | | | | | | | | | | | |

Note: * Located in on-chip RAM. The bus width is 32 bits when the DTC accesses this area as register information, and 16 bits otherwise.

| Address (low) | Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module Name | Data Bus Width |
|---|---|---|---|---|---|---|---|---|---|---|---|
| H'FE90 | TCR4 | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 | TPU4 | 16 bit |
| H'FE91 | TMDR4 | — | — | — | — | MD3 | MD2 | MD1 | MD0 | | |
| H'FE92 | TIOR4 | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 | | |
| H'FE94 | TIER4 | TTGE | — | TCIEU | TCIEV | — | — | TGIEB | TGIEA | | |
| H'FE95 | TSR4 | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA | | |
| H'FE96 | TCNT4 | | | | | | | | | | |
| H'FE97 | | | | | | | | | | | |
| H'FE98 | TGR4A | | | | | | | | | | |
| H'FE99 | | | | | | | | | | | |
| H'FE9A | TGR4B | | | | | | | | | | |
| H'FE9B | | | | | | | | | | | |
| H'FEA0 | TCR5 | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 | TPU5 | 16 bit |
| H'FEA1 | TMDR5 | — | — | — | — | MD3 | MD2 | MD1 | MD0 | | |
| H'FEA2 | TIOR5 | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 | | |
| H'FEA4 | TIER5 | TTGE | — | TCIEU | TCIEV | — | — | TGIEB | TGIEA | | |
| H'FEA5 | TSR5 | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA | | |
| H'FEA6 | TCNT5 | | | | | | | | | | |
| H'FEA7 | | | | | | | | | | | |
| H'FEA8 | TGR5A | | | | | | | | | | |
| H'FEA9 | | | | | | | | | | | |
| H'FEAA | TGR5B | | | | | | | | | | |
| H'FEAB | | | | | | | | | | | |
| H'FEB0 | P1DDR | P17DDR | P16DDR | P15DDR | P14DDR | P13DDR | P12DDR | P11DDR | P10DDR | Port | 8 bit |
| H'FEB1 | P2DDR | P27DDR | P26DDR | P25DDR | P24DDR | P23DDR | P22DDR | P21DDR | P20DDR | | |
| H'FEB2 | P3DDR | — | — | P35DDR | P34DDR | P33DDR | P32DDR | P31DDR | P30DDR | | |
| H'FEB9 | PADDR | — | — | — | — | PA3DDR | PA2DDR | PA1DDR | PA0DDR | | |
| H'FEBA | PBDDR | PB7DDR | PB6DDR | PB5DDR | PB4DDR | PB3DDR | PB2DDR | PB1DDR | PB0DDR | | |
| H'FEBB | PCDDR | PC7DDR | PC6DDR | PC5DDR | PC4DDR | PC3DDR | PC2DDR | PC1DDR | PC0DDR | | |
| H'FEBC | PDDDR | PD7DDR | PD6DDR | PD5DDR | PD4DDR | PD3DDR | PD2DDR | PD1DDR | PD0DDR | | |
| H'FEBD | PEDDR | PE7DDR | PE6DDR | PE5DDR | PE4DDR | PE3DDR | PE2DDR | PE1DDR | PE0DDR | | |
| H'FEBE | PFDDR | PF7DDR | PF6DDR | PF5DDR | PF4DDR | PF3DDR | PF2DDR | PF1DDR | PF0DDR | | |
| H'FEBF | PGDDR | — | — | — | PG4DDR | PG3DDR | PG2DDR | PG1DDR | PG0DDR | | |

| Address (low) | Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module Name | Data Bus Width |
|---|---|---|---|---|---|---|---|---|---|---|---|
| H'FEC4 | IPRA | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 | Interrupt controller | 8 bit |
| H'FEC5 | IPRB | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 | | |
| H'FEC6 | IPRC | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 | | |
| H'FEC7 | IPRD | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 | | |
| H'FEC8 | IPRE | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 | | |
| H'FEC9 | IPRF | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 | | |
| H'FECA | IPRG | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 | | |
| H'FECB | IPRH | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 | | |
| H'FECC | IPRI | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 | | |
| H'FECD | IPRJ | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 | | |
| H'FECE | IPRK | — | IPR6 | IPR5 | IPR4 | — | — | — | — | | |
| H'FED0 | ABWCR | ABW7 | ABW6 | ABW5 | ABW4 | ABW3 | ABW2 | ABW1 | ABW0 | Bus controller | 8 bit |
| H'FED1 | ASTCR | AST7 | AST6 | AST5 | AST4 | AST3 | AST2 | AST1 | AST0 | | |
| H'FED2 | WCRH | W71 | W70 | W61 | W60 | W51 | W50 | W41 | W40 | | |
| H'FED3 | WCRL | W31 | W30 | W21 | W20 | W11 | W10 | W01 | W00 | | |
| H'FED4 | BCRH | ICIS1 | ICIS0 | BRSTRM | BRSTS1 | BRSTS0 | — | — | — | | |
| H'FED5 | BCRL | BRLE | — | EAE | — | — | — | — | WAITE | | |
| H'FEDB | RAMER | — | — | — | — | — | RAMS | RAM1 | RAM0 | | |
| H'FF2C | ISCRH | IRQ7SCB | IRQ7SCA | IRQ6SCB | IRQ6SCA | IRQ5SCB | IRQ5SCA | IRQ4SCB | IRQ4SCA | Interrupt controller | 8 bit |
| H'FF2D | ISCRL | IRQ3SCB | IRQ3SCA | IRQ2SCB | IRQ2SCA | IRQ1SCB | IRQ1SCA | IRQ0SCB | IRQ0SCA | | |
| H'FF2E | IER | IRQ7E | IRQ6E | IRQ5E | IRQ4E | IRQ3E | IRQ2E | IRQ1E | IRQ0E | | |
| H'FF2F | ISR | IRQ7F | IRQ6F | IRQ5F | IRQ4F | IRQ3F | IRQ2F | IRQ1F | IRQ0F | | |
| H'FF30 to H'FF34 | DTCER | DTCE7 | DTCE6 | DTCE5 | DTCE4 | DTCE3 | DTCE2 | DTCE1 | DTCE0 | DTC | 8 bit |
| H'FF37 | DTVECR | SWDTE | DTVEC6 | DTVEC5 | DTVEC4 | DTVEC3 | DTVEC2 | DTVEC1 | DTVEC0 | | |
| H'FF38 | SBYCR | SSBY | STS2 | STS1 | STS0 | OPE | — | — | — | Power-down mode | 8 bit |
| H'FF39 | SYSCR | — | — | INTM1 | INTM0 | NMIEG | — | — | RAME | MCU | 8 bit |
| H'FF3A | SCKCR | PSTOP | — | — | — | — | SCK2 | SCK1 | SCK0 | Clock pulse generator | 8 bit |
| H'FF3B | MDCR | — | — | — | — | — | MDS2 | MDS1 | MDS0 | MCU | 8 bit |
| H'FF3C | MSTPCRH | MSTP15 | MSTP14 | MSTP13 | MSTP12 | MSTP11 | MSTP10 | MSTP9 | MSTP8 | Power-down mode | 8 bit |
| H'FF3D | MSTPCRL | MSTP7 | MSTP6 | MSTP5 | MSTP4 | MSTP3 | MSTP2 | MSTP1 | MSTP0 | | |
| H'FF42 | SYSCR2 | — | — | — | — | FLSHE | — | — | — | MCU | 8 bit |
| H'FF44 | Reserved | — | — | — | — | — | — | — | — | Reserved | — |
| H'FF45 | Reserved | — | — | — | — | — | — | — | — | | |

| Address (low) | Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module Name | Data Bus Width |
|---|---|---|---|---|---|---|---|---|---|---|---|
| H'FF50 | PORT1 | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 | Port | 8 bit |
| H'FF51 | PORT2 | P27 | P26 | P25 | P24 | P23 | P22 | P21 | P20 | | |
| H'FF52 | PORT3 | — | — | P35 | P34 | P33 | P32 | P31 | P30 | | |
| H'FF53 | PORT4 | P47 | P46 | P45 | P44 | P43 | P42 | P41 | P40 | | |
| H'FF59 | PORTA | — | — | — | — | PA3 | PA2 | PA1 | PA0 | | |
| H'FF5A | PORTB | PB7 | PB6 | PB5 | PB4 | PB3 | PB2 | PB1 | PB0 | | |
| H'FF5B | PORTC | PC7 | PC6 | PC5 | PC4 | PC3 | PC2 | PC1 | PC0 | | |
| H'FF5C | PORTD | PD7 | PD6 | PD5 | PD4 | PD3 | PD2 | PD1 | PD0 | | |
| H'FF5D | PORTE | PE7 | PE6 | PE5 | PE4 | PE3 | PE2 | PE1 | PE0 | | |
| H'FF5E | PORTF | PF7 | PF6 | PF5 | PF4 | PF3 | PF2 | PF1 | PF0 | | |
| H'FF5F | PORTG | — | — | — | PG4 | PG3 | PG2 | PG1 | PG0 | | |
| H'FF60 | P1DR | P17DR | P16DR | P15DR | P14DR | P13DR | P12DR | P11DR | P10DR | | |
| H'FF61 | P2DR | P27DR | P26DR | P25DR | P24DR | P23DR | P22DR | P21DR | P20DR | | |
| H'FF62 | P3DR | — | — | P35DR | P34DR | P33DR | P32DR | P31DR | P30DR | | |
| H'FF69 | PADR | — | — | — | — | PA3DR | PA2DR | PA1DR | PA0DR | | |
| H'FF6A | PBDR | PB7DR | PB6DR | PB5DR | PB4DR | PB3DR | PB2DR | PB1DR | PB0DR | | |
| H'FF6B | PCDR | PC7DR | PC6DR | PC5DR | PC4DR | PC3DR | PC2DR | PC1DR | PC0DR | | |
| H'FF6C | PDDR | PD7DR | PD6DR | PD5DR | PD4DR | PD3DR | PD2DR | PD1DR | PD0DR | | |
| H'FF6D | PEDR | PE7DR | PE6DR | PE5DR | PE4DR | PE3DR | PE2DR | PE1DR | PE0DR | | |
| H'FF6E | PFDR | PF7DR | PF6DR | PF5DR | PF4DR | PF3DR | PF2DR | PF1DR | PF0DR | | |
| H'FF6F | PGDR | — | — | — | PG4DR | PG3DR | PG2DR | PG1DR | PG0DR | | |
| H'FF70 | PAPCR | — | — | — | — | PA3PCR | PA2PCR | PA1PCR | PA0PCR | | |
| H'FF71 | PBPCR | PB7PCR | PB6PCR | PB5PCR | PB4PCR | PB3PCR | PB2PCR | PB1PCR | PB0PCR | | |
| H'FF72 | PCPCR | PC7PCR | PC6PCR | PC5PCR | PC4PCR | PC3PCR | PC2PCR | PC1PCR | PC0PCR | | |
| H'FF73 | PDPCR | PD7PCR | PD6PCR | PD5PCR | PD4PCR | PD3PCR | PD2PCR | PD1PCR | PD0PCR | | |
| H'FF74 | PEPCR | PE7PCR | PE6PCR | PE5PCR | PE4PCR | PE3PCR | PE2PCR | PE1PCR | PE0PCR | | |
| H'FF76 | P3ODR | — | — | P35ODR | P34ODR | P33ODR | P32ODR | P31ODR | P30ODR | | |
| H'FF77 | PAODR | — | — | — | — | PA3ODR | PA2ODR | PA1ODR | PA0ODR | | |

| Address (low) | Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module Name | Data Bus Width |
|---|---|---|---|---|---|---|---|---|---|---|---|
| H'FF78 | SMR0 | C/$\overline{A}$/ GM*[1] | CHR | PE | O/$\overline{E}$ | STOP | MP | CKS1 | CKS0 | SCI0, Smart card interface 0 | 8 bit |
| H'FF79 | BRR0 | | | | | | | | | | |
| H'FF7A | SCR0 | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 | | |
| H'FF7B | TDR0 | | | | | | | | | | |
| H'FF7C | SSR0 | TDRE | RDRF | ORER | FER/ ERS*[2] | PER | TEND | MPB | MPBT | | |
| H'FF7D | RDR0 | | | | | | | | | | |
| H'FF7E | SCMR0 | — | — | — | — | SDIR | SINV | — | SMIF | | |
| H'FF80 | SMR1 | C/$\overline{A}$/ GM*[1] | CHR | PE | O/$\overline{E}$ | STOP | MP | CKS1 | CKS0 | SCI1, Smart card interface 1 | 8 bit |
| H'FF81 | BRR1 | | | | | | | | | | |
| H'FF82 | SCR1 | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 | | |
| H'FF83 | TDR1 | | | | | | | | | | |
| H'FF84 | SSR1 | TDRE | RDRF | ORER | FER/ ERS*[2] | PER | TEND | MPB | MPBT | | |
| H'FF85 | RDR1 | | | | | | | | | | |
| H'FF86 | SCMR1 | — | — | — | — | SDIR | SINV | — | SMIF | | |
| H'FF90 | ADDRAH | AD9 | AD8 | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | A/D converter | 8 bit |
| H'FF91 | ADDRAL | AD1 | AD0 | — | — | — | — | — | — | | |
| H'FF92 | ADDRBH | AD9 | AD8 | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | | |
| H'FF93 | ADDRBL | AD1 | AD0 | — | — | — | — | — | — | | |
| H"FF94 | ADDRCH | AD9 | AD8 | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | | |
| H'FF95 | ADDRCL | AD1 | AD0 | — | — | — | — | — | — | | |
| H'FF96 | ADDRDH | AD9 | AD8 | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | | |
| H'FF97 | ADDRDL | AD1 | AD0 | — | — | — | — | — | — | | |
| H'FF98 | ADCSR | ADF | ADIE | ADST | SCAN | CKS | — | CH1 | CH0 | | |
| H'FF99 | ADCR | TRGS1 | TRGS0 | — | — | — | — | — | — | | |

Notes: 1. Functions as C/$\overline{A}$ for SCI use, and as GM for smart card interface use.
2. Functions as FER for SCI use, and as ERS for smart card interface use.

| Address (low) | Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module Name | Data Bus Width |
|---|---|---|---|---|---|---|---|---|---|---|---|
| H'FFA4 | DADR0 | | | | | | | | | D/A converter | 8 bit |
| H'FFA5 | DADR1 | | | | | | | | | | |
| H'FFA6 | DACR | DAOE1 | DAOE0 | DAE | — | — | — | — | — | | |
| H'FFB0 | TCR0 | CMIEB | CMIEA | OVIE | CCLR1 | CCLR0 | CKS2 | CKS1 | CKS0 | 8-bit timer channel 0, 1 | 16 bit |
| H'FFB1 | TCR1 | CMIEB | CMIEA | OVIE | CCLR1 | CCLR0 | CKS2 | CKS1 | CKS0 | | |
| H'FFB2 | TCSR0 | CMFB | CMFA | OVF | ADTE | OS3 | OS2 | OS1 | OS0 | | |
| H'FFB3 | TCSR1 | CMFB | CMFA | OVF | — | OS3 | OS2 | OS1 | OS0 | | |
| H'FFB4 | TCORA0 | | | | | | | | | | |
| H'FFB5 | TCORA1 | | | | | | | | | | |
| H'FFB6 | TCORB0 | | | | | | | | | | |
| H'FFB7 | TCORB1 | | | | | | | | | | |
| H'FFB8 | TCNT0 | | | | | | | | | | |
| H'FFB9 | TCNT1 | | | | | | | | | | |
| H'FFBC (read) | TCSR | OVF | WT/$\overline{IT}$ | TME | — | — | CKS2 | CKS1 | CKS0 | WDT | 16 bit |
| H'FFBD (read) | TCNT | | | | | | | | | | |
| H'FFBF (read) | RSTCSR | WOVF | RSTE | RSTS | — | — | — | — | — | | |
| H'FFC0 | TSTR | — | — | CST5 | CST4 | CST3 | CST2 | CST1 | CST0 | TPU | 16 bit |
| H'FFC1 | TSYR | — | — | SYNC5 | SYNC4 | SYNC3 | SYNC2 | SYNC1 | SYNC0 | | |
| H'FFC8 | FLMCR1 | FWE | SWE | — | — | EV | PV | E | P | FLASH | 8 bit |
| H'FFC9 | FLMCR2 | FLER | — | — | — | — | — | ESU | PSU | | |
| H'FFCA | EBR1 | — | — | — | — | — | — | EB9 | EB8 | | |
| H'FFCB | EBR2 | EB7 | EB6 | EB5 | EB4 | EB3 | EB2 | EB1 | EB0 | | |
| H'FFD0 | TCR0 | CCLR2 | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 | TPU0 | 16 bit |
| H'FFD1 | TMDR0 | — | — | BFB | BFA | MD3 | MD2 | MD1 | MD0 | | |
| H'FFD2 | TIOR0H | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 | | |
| H'FFD3 | TIOR0L | IOD3 | IOD2 | IOD1 | IOD0 | IOC3 | IOC2 | IOC1 | IOC0 | | |
| H'FFD4 | TIER0 | TTGE | — | — | TCIEV | TGIED | TGIEC | TGIEB | TGIEA | | |
| H'FFD5 | TSR0 | — | — | — | TCFV | TGFD | TGFC | TGFB | TGFA | | |
| H'FFD6 | TCNT0 | | | | | | | | | | |
| H'FFD7 | | | | | | | | | | | |
| H'FFD8 | TGR0A | | | | | | | | | | |
| H'FFD9 | | | | | | | | | | | |
| H'FFDA | TGR0B | | | | | | | | | | |
| H'FFDB | | | | | | | | | | | |

| Address (low) | Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module Name | Data Bus Width |
|---|---|---|---|---|---|---|---|---|---|---|---|
| H'FFDC | TGR0C | | | | | | | | | TPU1 | 16 bit |
| H'FFDD | | | | | | | | | | | |
| H'FFDE | TGR0D | | | | | | | | | | |
| H'FFDF | | | | | | | | | | | |
| H'FFE0 | TCR1 | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 | | |
| H'FFE1 | TMDR1 | — | — | — | — | MD3 | MD2 | MD1 | MD0 | | |
| H'FFE2 | TIOR1 | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 | | |
| H'FFE4 | TIER1 | TTGE | — | TCIEU | TCIEV | — | — | TGIEB | TGIEA | | |
| H'FFE5 | TSR1 | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA | | |
| H'FFE6 | TCNT1 | | | | | | | | | | |
| H'FFE7 | | | | | | | | | | | |
| H'FFE8 | TGR1A | | | | | | | | | | |
| H'FFE9 | | | | | | | | | | | |
| H'FFEA | TGR1B | | | | | | | | | | |
| H'FFEB | | | | | | | | | | | |
| H'FFF0 | TCR2 | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 | TPU2 | 16 bit |
| H'FFF1 | TMDR2 | — | — | — | — | MD3 | MD2 | MD1 | MD0 | | |
| H'FFF2 | TIOR2 | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 | | |
| H'FFF4 | TIER2 | TTGE | — | TCIEU | TCIEV | — | — | TGIEB | TGIEA | | |
| H'FFF5 | TSR2 | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA | | |
| H'FFF6 | TCNT2 | | | | | | | | | | |
| H'FFF7 | | | | | | | | | | | |
| H'FFF8 | TGR2A | | | | | | | | | | |
| H'FFF9 | | | | | | | | | | | |
| H'FFFA | TGR2B | | | | | | | | | | |
| H'FFFB | | | | | | | | | | | |

**MRA—DTC Mode Register A** **H'F800—H'FBFF** **DTC**

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SM1 | SM0 | DM1 | DM0 | MD1 | MD0 | DTS | Sz |
| Initial value : | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined |
| Read/Write : | — | — | — | — | — | — | — | — |

DTC Data Transfer Size

| | |
|---|---|
| 0 | Byte-size transfer |
| 1 | Word-size transfer |

DTC Transfer Mode Select

| | |
|---|---|
| 0 | Destination side is repeat area or block area |
| 1 | Source side is repeat area or block area |

DTC Mode

| | | |
|---|---|---|
| 0 | 0 | Normal mode |
| | 1 | Repeat mode |
| 1 | 0 | Block transfer mode |
| | 1 | — |

Destination Address Mode

| | | |
|---|---|---|
| 0 | — | DAR is fixed |
| 1 | 0 | DAR is incremented after a transfer (by +1 when Sz = 0; by +2 when Sz = 1) |
| | 1 | DAR is decremented after a transfer (by -1 when Sz = 0; by -2 when Sz = 1) |

Source Address Mode

| | | |
|---|---|---|
| 0 | — | SAR is fixed |
| 1 | 0 | SAR is incremented after a transfer (by +1 when Sz = 0; by +2 when Sz = 1) |
| | 1 | SAR is decremented after a transfer (by -1 when Sz = 0; by -2 when Sz = 1) |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CHNE | DISEL | — | — | — | — | — | — |
| Initial value | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined |
| Read/Write | — | — | — | — | — | — | — | — |

Reserved
Only 0 should be written to these bits

DTC Interrupt Select

| | |
|---|---|
| 0 | After a data transfer ends, the CPU interrupt is disabled unless the transfer counter is 0 |
| 1 | After a data transfer ends, the CPU interrupt is enabled |

DTC Chain Transfer Enable

| | |
|---|---|
| 0 | End of DTC data transfer |
| 1 | DTC chain transfer |

**SAR—DTC Source Address Register** **H'F800—H'FBFF** **DTC**

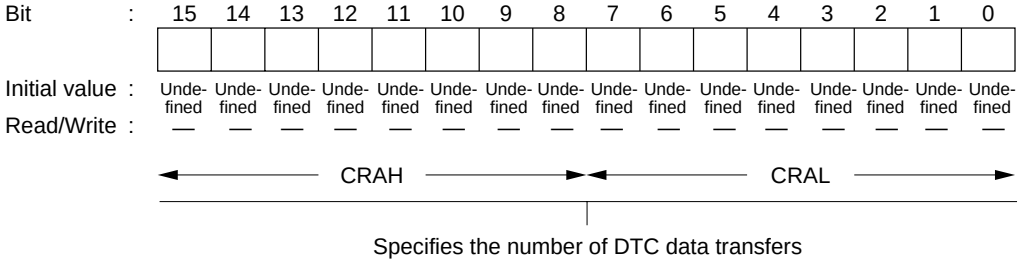
| Bit | 23 | 22 | 21 | 20 | 19 | - - - | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | - - - | | | | | |
| Initial value | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | - - - | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined |
| Read/Write | — | — | — | — | — | - - - | — | — | — | — | — |

Specifies transfer data source address

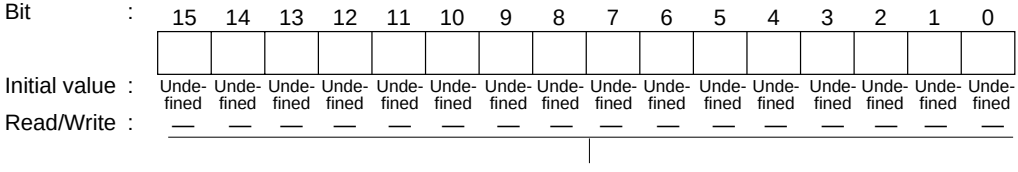
**DAR—DTC Destination Address Register** **H'F800—H'FBFF** **DTC**

| Bit | 23 | 22 | 21 | 20 | 19 | - - - | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | - - - | | | | | |
| Initial value | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined | - - - | Unde-fined | Unde-fined | Unde-fined | Unde-fined | Unde-fined |
| Read/Write | — | — | — | — | — | - - - | — | — | — | — | — |

Specifies transfer data destination address




**CRB—DTC Transfer Count Register B** **H'F800—H'FBFF** **DTC**



| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | CCLR2 | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Timer Prescaler

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | Internal clock: counts on ø/1 |
| | | 1 | Internal clock: counts on ø/4 |
| | 1 | 0 | Internal clock: counts on ø/16 |
| | | 1 | Internal clock: counts on ø/64 |
| 1 | 0 | 0 | External clock: counts on TCLKA pin input |
| | | 1 | Internal clock: counts on ø/1024 |
| | 1 | 0 | Internal clock: counts on ø/256 |
| | | 1 | Internal clock: counts on ø/4096 |

Clock Edge

| | | |
|---|---|---|
| 0 | 0 | Count at rising edge |
| | 1 | Count at falling edge |
| 1 | — | Count at both edges |

Counter Clear

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | TCNT clearing disabled |
| | | 1 | TCNT cleared by TGRA compare match/input capture |
| | 1 | 0 | TCNT cleared by TGRB compare match/input capture |
| | | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation *1 |
| 1 | 0 | 0 | TCNT clearing disabled |
| | | 1 | TCNT cleared by TGRC compare match/input capture *2 |
| | 1 | 0 | TCNT cleared by TGRD compare match/input capture *2 |
| | | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation *1 |

Notes: 1. Synchronous operation setting is performed by setting the SYNC bit in TSYR to 1.
2. When TGRC or TGRD is used as a buffer register, TCNT is not cleared because the buffer register setting has priority, and compare match/input capture does not occur.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | BFB | BFA | MD3 | MD2 | MD1 | MD0 |
| Initial value | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

Mode

| | | | | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Normal operation |
| | | | 1 | Reserved |
| | | 1 | 0 | PWM mode 1 |
| | | | 1 | PWM mode 2 |
| | 1 | 0 | 0 | Phase counting mode 1 |
| | | | 1 | Phase counting mode 2 |
| | | 1 | 0 | Phase counting mode 3 |
| | | | 1 | Phase counting mode 4 |
| 1 | * | * | * | — |

* : Don't care

Notes: 1. MD3 is a reserved bit. In a write, it should always be written with 0.
2. Phase counting mode cannot be set for channels 0 and 3. In this case, 0 should always be written to MD2.

Buffer Operation A

| | |
|---|---|
| 0 | TGRA operates normally |
| 1 | TGRA and TGRC used together for buffer operation |

Buffer Operation B

| | |
|---|---|
| 0 | TGRB operates normally |
| 1 | TGRB and TGRD used together for buffer operation |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Initial value | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TGR3A I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR3A is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR3A is input capture register | Capture input source is TIOCA3 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is channel 4/count clock | Input capture at TCNT4 count-up/ count-down |

* : Don't care

TGR3B I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR3B is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR3B is input capture register | Capture input source is TIOCB3 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is channel 4/count clock | Input capture at TCNT4 count-up/ count-down |

* : Don't care

Note: 1. If bits TPSC2 to TPSC0 in TCR4 are set to B'000, and ø/1 is used as the TCNT4 count clock, this setting will be invalid and input capture will not occur.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | IOD3 | IOD2 | IOD1 | IOD0 | IOC3 | IOC2 | IOC1 | IOC0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TRG3C I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR3C is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR3C is input capture register | Capture input source is TIOCC3 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is channel 4/count clock | Input capture at TCNT4 count-up/ count-down |

* : Don't care

Note: When the BFA bit in TMDR3 is set to 1 and TGR3C is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

TGR3D I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR3D is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR3D is input capture register | Capture input source is TIOCD3 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is channel 4/count clock | Input capture at TCNT4 count-up/ count-down*1 |

* : Don't care

Notes: When the BFB bit in TMDR3 is set to 1 and TGR3D is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

1 When bits TPSC2 to TPSC0 in TCR4 are set to B'000 and ø/1 is used as the TCNT4 count clock, this setting is invalid and input capture is not generated.

Note: When TGRC or TGRD is designated for buffer operation, this setting is invalid and the register operates as a buffer register.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TTGE | — | — | TCIEV | TGIED | TGIEC | TGIEB | TGIEA |
| Initial value | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | — | — | R/W | R/W | R/W | R/W | R/W |

TGR Interrupt Enable A

| | |
|---|---|
| 0 | Interrupt requests (TGIA) by TGFA bit disabled |
| 1 | Interrupt requests (TGIA) by TGFA bit enabled |

TGR Interrupt Enable B

| | |
|---|---|
| 0 | Interrupt requests (TGIB) by TGFB bit disabled |
| 1 | Interrupt requests (TGIB) by TGFB bit enabled |

TGR Interrupt Enable C

| | |
|---|---|
| 0 | Interrupt requests (TGIC) by TGFC bit disabled |
| 1 | Interrupt requests (TGIC) by TGFC bit enabled |

TGR Interrupt Enable D

| | |
|---|---|
| 0 | Interrupt requests (TGID) by TGFD bit disabled |
| 1 | Interrupt requests (TGID) by TGFD bit enabled |

Overflow Interrupt Enable

| | |
|---|---|
| 0 | Interrupt requests (TCIV) by TCFV disabled |
| 1 | Interrupt requests (TCIV) by TCFV enabled |

A/D Conversion Start Request Enable

| | |
|---|---|
| 0 | A/D conversion start request generation disabled |
| 1 | A/D conversion start request generation enabled |

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | TCFV | TGFD | TGFC | TGFB | TGFA |
| Initial value : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | — | — | — | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* |

Input Capture/Output Compare Flag A

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIA interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFA after reading TGFA = 1 |
| 1 | [Setting condition]<br>• When TCNT=TGRA while TGRA is functioning as output compare register<br>• When TCNT value is transferred to TGRA by input capture signal while TGRA is functioning as input capture register |

Input Capture/Output Compare Flag B

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIB interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFB after reading TGFB = 1 |
| 1 | [Setting condition]<br>• When TCNT = TGRB while TGRB is functioning as output compare register<br>• When TCNT value is transferred to TGRB by input capture signal while TGRB is functioning as input capture register |

Input Capture/Output Compare Flag C

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIC interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFC after reading TGFC = 1 |
| 1 | [Setting condition]<br>• When TCNT = TGRC while TGRC is functioning as output compare register<br>• When TCNT value is transferred to TGRC by input capture signal while TGRC is functioning as input capture register |

Input Capture/Output Compare Flag D

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGID interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFD after reading TGFD = 1 |
| 1 | [Setting condition]<br>• When TCNT = TGRD while TGRD is functioning as output compare register<br>• When TCNT value is transferred to TGRD by input capture signal while TGRD is functioning as input capture register |

Overflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFV after reading TCFV = 1 |
| 1 | [Setting condition]<br>When the TCNT value overflows (changes from H'FFFF to H'0000 ) |

Note: * Can only be written with 0 for flag clearing.

**TCNT3—Timer Counter 3** **H'FE86** **TPU3**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Up-counter

**TGR3A—Timer General Register 3A** **H'FE88** **TPU3**
**TGR3B—Timer General Register 3B** **H'FE8A** **TPU3**
**TGR3C—Timer General Register 3C** **H'FE8C** **TPU3**
**TGR3D—Timer General Register 3D** **H'FE8E** **TPU3**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Initial value | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Timer Prescaler

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | Internal clock: counts on ø/1 |
| | | 1 | Internal clock: counts on ø/4 |
| | 1 | 0 | Internal clock: counts on ø/16 |
| | | 1 | Internal clock: counts on ø/64 |
| 1 | 0 | 0 | External clock: counts on TCLKA pin input |
| | | 1 | External clock: counts on TCLKC pin input |
| | 1 | 0 | Internal clock: counts on ø/1024 |
| | | 1 | Counts on TCNT5 overflow/underflow |

Note: This setting is ignored when channel 4 is in phase counting mode.

Clock Edge

| | | |
|---|---|---|
| 0 | 0 | Count at rising edge |
| | 1 | Count at falling edge |
| 1 | — | Count at both edges |

Note: This setting is ignored when channel 4 is in phase counting mode.

Counter Clear

| | | |
|---|---|---|
| 0 | 0 | TCNT clearing disabled |
| | 1 | TCNT cleared by TGRA compare match/input capture |
| 1 | 0 | TCNT cleared by TGRB compare match/input capture |
| | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation* |

Note: * Synchronous operating setting is performed by setting the SYNC bit TSYR to 1.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | MD3 | MD2 | MD1 | MD0 |
| Initial value | : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | — | — | — | — | R/W | R/W | R/W | R/W |

Mode

| | | | | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Normal operation |
| | | | 1 | Reserved |
| | | 1 | 0 | PWM mode 1 |
| | | | 1 | PWM mode 2 |
| | 1 | 0 | 0 | Phase counting mode 1 |
| | | | 1 | Phase counting mode 2 |
| | | 1 | 0 | Phase counting mode 3 |
| | | | 1 | Phase counting mode 4 |
| 1 | * | * | * | — |

* : Don't care

Notes: MD3 is a reserved bit. In a write, it should always be written with 0.

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TGR4A I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR4A is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR4A is input capture register | Capture input source is TIOCA4 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is TGR3A compare match/ input capture | Input capture at generation of TGR3A compare match/input capture |

* : Don't care

TGR4B I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR4B is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR4B is input capture register | Capture input source is TIOCB4 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is TGR3C compare match/ input capture | Input capture at generation of TGR3C compare match/input capture |

* : Don't care

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TTGE | — | TCIEU | TCIEV | — | — | TGIEB | TGIEA |
| Initial value | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | — | R/W | R/W | — | — | R/W | R/W |

TGR Interrupt Enable A

| | |
|---|---|
| 0 | Interrupt requests (TGIA) by TGFA bit disabled |
| 1 | Interrupt requests (TGIA) by TGFA bit enabled |

TGR Interrupt Enable B

| | |
|---|---|
| 0 | Interrupt requests (TGIB) by TGFB bit disabled |
| 1 | Interrupt requests (TGIB) by TGFB bit enabled |

Overflow Interrupt Enable

| | |
|---|---|
| 0 | Interrupt requests (TCIV) by TCFV disabled |
| 1 | Interrupt requests (TCIV) by TCFV enabled |

Underflow Interrupt Enable

| | |
|---|---|
| 0 | Interrupt requests (TCIU) by TCFU disabled |
| 1 | Interrupt requests (TCIU) by TCFU enabled |

A/D Conversion Start Request Enable

| | |
|---|---|
| 0 | A/D conversion start request generation disabled |
| 1 | A/D conversion start request generation enabled |

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA |
| Initial value : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R | — | R/(W)* | R/(W)* | — | — | R/(W)* | R/(W)* |

Input Capture/Output Compare Flag A

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIA interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFA after reading TGFA = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRA while TGRA is functioning as output compare register<br>• When TCNT value is transferred to TGRA by input capture signal while TGRA is functioning as input capture register |

Input Capture/Output Compare Flag B

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIB interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFB after reading TGFB = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRB while TGRB is functioning as output compare register<br>• When TCNT value is transferred to TGRB by input capture signal while TGRB is functioning as input capture register |

Overflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFV after reading TCFV = 1 |
| 1 | [Setting conditions]<br>When the TCNT value overflows (changes from H'FFFF to H'0000 ) |

Underflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFU after reading TCFU = 1 |
| 1 | [Setting conditions]<br>When the TCNT value underflows (changes from H'0000 to H'FFFF) |

Count Direction Flag

| | |
|---|---|
| 0 | TCNT counts down |
| 1 | TCNT counts up |

Note: * Can only be written with 0 for flag clearing.

**TCNT4—Timer Counter 4** **H'FE96** **TPU4**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Up/down-counter*

Note: * This timer counter can be used as an up/down-counter only in phase counting mode or when performing overflow/underflow counting on another channel. In other cases it functions as an up-counter.

**TGR4A—Timer General Register 4A** **H'FE98** **TPU4**
**TGR4B—Timer General Register 4B** **H'FE9A** **TPU4**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Time Prescaler

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | Internal clock: counts on ø/1 |
| | | 1 | Internal clock: counts on ø/4 |
| | 1 | 0 | Internal clock: counts on ø/16 |
| | | 1 | Internal clock: counts on ø/64 |
| 1 | 0 | 0 | External clock: counts on TCLKA pin input |
| | | 1 | External clock: counts on TCLKC pin input |
| | 1 | 0 | Internal clock: counts on ø/256 |
| | | 1 | External clock: counts on TCLKD pin input |

Note: This setting is ignored when channel 5 is in phase counting mode.

Clock Edge

| | | |
|---|---|---|
| 0 | 0 | Count at rising edge |
| | 1 | Count at falling edge |
| 1 | — | Count at both edges |

Note: This setting is ignored when channel 5 is in phase counting mode.

Counter Clear

| | | |
|---|---|---|
| 0 | 0 | TCNT clearing disabled |
| | 1 | TCNT cleared by TGRA compare match/input capture |
| 1 | 0 | TCNT cleared by TGRB compare match/input capture |
| | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation* |

Note: * Synchronous operating setting is performed by setting the SYNC bit TSYR to 1.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | MD3 | MD2 | MD1 | MD0 |
| Initial value | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | — | — | R/W | R/W | R/W | R/W |

Mode

| | | | | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Normal operation |
| | | | 1 | Reserved |
| | | 1 | 0 | PWM mode 1 |
| | | | 1 | PWM mode 2 |
| | 1 | 0 | 0 | Phase counting mode 1 |
| | | | 1 | Phase counting mode 2 |
| | | 1 | 0 | Phase counting mode 3 |
| | | | 1 | Phase counting mode 4 |
| 1 | * | * | * | — |

* : Don't care

Notes: MD3 is a reserved bit. In a write, it should always be written with 0.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TGR5A I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR5A is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | * | 0 | 0 | TGR5A is input capture register | Capture input source is TIOCA5 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |

* : Don’t care

TGR5B I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR5B is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | * | 0 | 0 | TGR5B is input capture register | Capture input source is TIOCB5 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |

* : Don’t care

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TTGE | — | TCIEU | TCIEV | — | — | TGIEB | TGIEA |
| Initial value | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | — | R/W | R/W | — | — | R/W | R/W |

TGR Interrupt Enable A

| | |
|---|---|
| 0 | Interrupt requests (TGIA) by TGFA bit disabled |
| 1 | Interrupt requests (TGIA) by TGFA bit enabled |

TGR Interrupt Enable B

| | |
|---|---|
| 0 | Interrupt requests (TGIB) by TGFB bit disabled |
| 1 | Interrupt requests (TGIB) by TGFB bit enabled |

Overflow Interrupt Enable

| | |
|---|---|
| 0 | Interrupt requests (TCIV) by TCFV disabled |
| 1 | Interrupt requests (TCIV) by TCFV enabled |

Underflow Interrupt Enable

| | |
|---|---|
| 0 | Interrupt requests (TCIU) by TCFU disabled |
| 1 | Interrupt requests (TCIU) by TCFU enabled |

A/D Conversion Start Request Enable

| | |
|---|---|
| 0 | A/D conversion start request generation disabled |
| 1 | A/D conversion start request generation enabled |

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA |
| Initial value : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R | — | R/(W)* | R/(W)* | — | — | R/(W)* | R/(W)* |

Input Capture/Output Compare Flag A

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIA interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFA after reading TGFA = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRA while TGRA is functioning as output compare register<br>• When TCNT value is transferred to TGRA by input capture signal while TGRA is functioning as input capture register |

Input Capture/Output Compare Flag B

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIB interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFB after reading TGFB = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRB while TGRB is functioning as output compare register<br>• When TCNT value is transferred to TGRB by input capture signal while TGRB is functioning as input capture register |

Overflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFV after reading TCFV = 1 |
| 1 | [Setting conditions]<br>When the TCNT value overflows (changes from H'FFFF to H'0000 ) |

Underflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFU after reading TCFU = 1 |
| 1 | [Setting conditions]<br>When the TCNT value underflows (changes from H'0000 to H'FFFF) |

Count Direction Flag

| | |
|---|---|
| 0 | TCNT counts down |
| 1 | TCNT counts up |

Note: * Can only be written with 0 for flag clearing.

**TCNT5—Timer Counter 5** **H'FEA6** **TPU5**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Up/down-counter*

Note: * This timer counter can be used as an up/down-counter only in phase counting mode or when performing overflow/underflow counting on another channel. In other cases it functions as an up-counter.

**TGR5A—Timer General Register 5A** **H'FEA8** **TPU5**
**TGR5B—Timer General Register 5B** **H'FEAA** **TPU5**

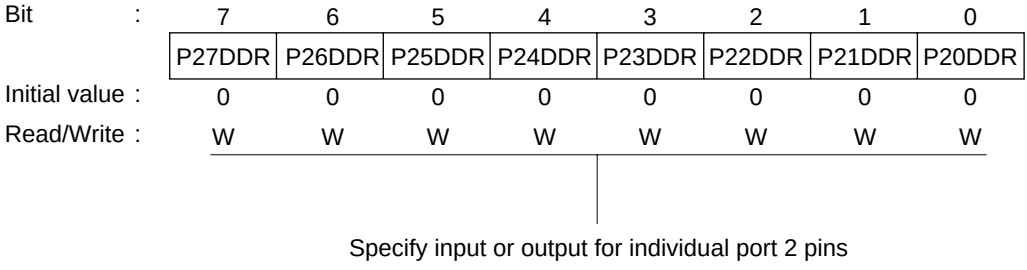
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Initial value | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**P1DDR—Port 1 Data Direction Register** **H'FEB0** **Port 1**

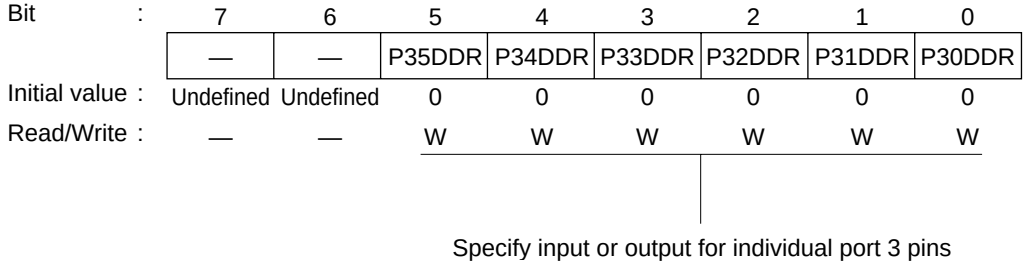
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P17DDR | P16DDR | P15DDR | P14DDR | P13DDR | P12DDR | P11DDR | P10DDR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | W | W | W | W | W | W | W | W |

Specify input or output for individual port 1 pins

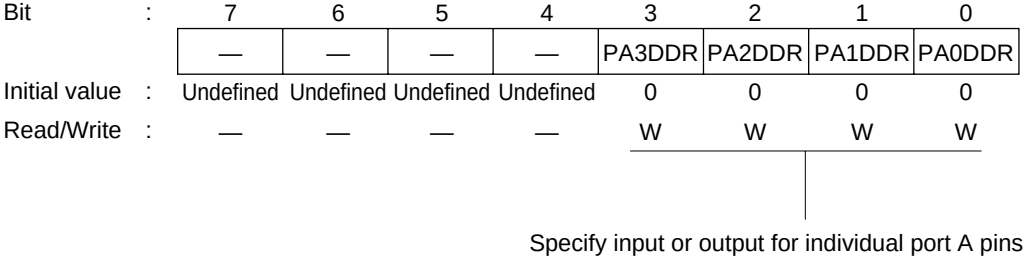
**P2DDR—Port 2 Data Direction Register** **H'FEB1** **Port 2**



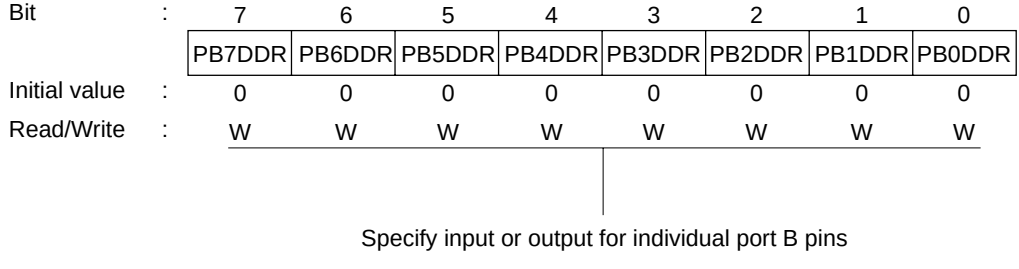

**P3DDR—Port 3 Data Direction Register** **H'FEB2** **Port 3**




**PADDR—Port A Data Direction Register** **H'FEB9** **Port A**




**PBDDR—Port B Data Direction Register** **H'FEBA** **Port B**



**PCDDR—Port C Data Direction Register H'FEBB Port C**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PC7DDR | PC6DDR | PC5DDR | PC4DDR | PC3DDR | PC2DDR | PC1DDR | PC0DDR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | W | W | W | W | W | W | W | W |

Specify input or output for individual port C pins

**PDDDR—Port D Data Direction Register H'FEBC Port D**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PD7DDR | PD6DDR | PD5DDR | PD4DDR | PD3DDR | PD2DDR | PD1DDR | PD0DDR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | W | W | W | W | W | W | W | W |

Specify input or output for individual port D pins

**PEDDR—Port E Data Direction Register H'FEBD Port E**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PE7DDR | PE6DDR | PE5DDR | PE4DDR | PE3DDR | PE2DDR | PE1DDR | PE0DDR |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | W | W | W | W | W | W | W | W |

Specify input or output for individual port E pins

**PFDDR—Port F Data Direction Register** **H'FEBE** **Port F**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | PF7DDR | PF6DDR | PF5DDR | PF4DDR | PF3DDR | PF2DDR | PF1DDR | PF0DDR |
| Modes 1, 2, 4 to 6 | | | | | | | | | |
| Initial value | : | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | W | W | W | W | W | W | W | W |
| Modes 3, 7 | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | W | W | W | W | W | W | W | W |

Specify input or output for individual port F pins

**PGDDR—Port G Data Direction Register** **H'FEBF** **Port G**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | PG4DDR | PG3DDR | PG2DDR | PG1DDR | PG0DDR |
| Modes 1, 4, 5 | | | | | | | | | |
| Initial value | : | Undefined | Undefined | Undefined | 1 | 0 | 0 | 0 | 0 |
| Read/Write | : | — | — | — | W | W | W | W | W |
| Modes 2, 3, 6, 7 | | | | | | | | | |
| Initial value | : | Undefined | Undefined | Undefined | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | — | — | — | W | W | W | W | W |

Specify input or output for individual port G pins

| | | | |
|---|---|---|---|
| **IPRA — Interrupt Priority Register A** | **H'FEC4** | **Interrupt Controller** |
| **IPRB — Interrupt Priority Register B** | **H'FEC5** | **Interrupt Controller** |
| **IPRC — Interrupt Priority Register C** | **H'FEC6** | **Interrupt Controller** |
| **IPRD — Interrupt Priority Register D** | **H'FEC7** | **Interrupt Controller** |
| **IPRE — Interrupt Priority Register E** | **H'FEC8** | **Interrupt Controller** |
| **IPRF — Interrupt Priority Register F** | **H'FEC9** | **Interrupt Controller** |
| **IPRG — Interrupt Priority Register G** | **H'FECA** | **Interrupt Controller** |
| **IPRH — Interrupt Priority Register H** | **H'FECB** | **Interrupt Controller** |
| **IPRI — Interrupt Priority Register I** | **H'FECC** | **Interrupt Controller** |
| **IPRJ — Interrupt Priority Register J** | **H'FECD** | **Interrupt Controller** |
| **IPRK — Interrupt Priority Register K** | **H'FECE** | **Interrupt Controller** |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | IPR6 | IPR5 | IPR4 | — | IPR2 | IPR1 | IPR0 |
| Initial value | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| Read/Write | — | R/W | R/W | R/W | — | R/W | R/W | R/W |

Set priority (levels 7 to 0) for interrupt sources

Correspondence between Interrupt Sources and IPR Settings

| Register | Bits | |
|---|---|---|
| | 6 to 4 | 2 to 0 |
| IPRA | IRQ0 | IRQ1 |
| IPRB | IRQ2<br>IRQ3 | IRQ4<br>IRQ5 |
| IPRC | IRQ6<br>IRQ7 | DTC |
| IPRD | WDT | —* |
| IPRE | —* | A/D converter |
| IPRF | TPU channel 0 | TPU channel 1 |
| IPRG | TPU channel 2 | TPU channel 3 |
| IPRH | TPU channel 4 | TPU channel 5 |
| IPRI | 8-bit timer channel 0 | 8-bit timer channel 1 |
| IPRJ | —* | SCI channel 0 |
| IPRK | SCI channel 1 | —* |

Note: * Reserved bits. May be read or written, but the setting is ignored.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | ABW7 | ABW6 | ABW5 | ABW4 | ABW3 | ABW2 | ABW1 | ABW0 |
| Modes 1 to 3, 5 to 7 | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Mode 4 | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Area 7 to 0 Bus Width Control

| | |
|---|---|
| 0 | Area n is designated for 16-bit access |
| 1 | Area n is designated for 8-bit access |

(n = 7 to 0)

**ASTCR—Access State Control Register** **H'FED1** **Bus Controller**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | AST7 | AST6 | AST5 | AST4 | AST3 | AST2 | AST1 | AST0 |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Area 7 to 0 Access State Control

| | |
|---|---|
| 0 | Area n is designated for 2-state access<br>Wait state insertion in area n external space is disabled |
| 1 | Area n is designated for 3-state access<br>Wait state insertion in area n external space is enabled |

(n = 7 to 0)

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | W71 | W70 | W61 | W60 | W51 | W50 | W41 | W40 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Area 4 Wait Control

| | | |
|---|---|---|
| 0 | 0 | Program wait not inserted |
| | 1 | 1 program wait state inserted |
| 1 | 0 | 2 program wait states inserted |
| | 1 | 3 program wait states inserted |

Area 5 Wait Control

| | | |
|---|---|---|
| 0 | 0 | Program wait not inserted |
| | 1 | 1 program wait state inserted |
| 1 | 0 | 2 program wait states inserted |
| | 1 | 3 program wait states inserted |

Area 6 Wait Control

| | | |
|---|---|---|
| 0 | 0 | Program wait not inserted |
| | 1 | 1 program wait state inserted |
| 1 | 0 | 2 program wait states inserted |
| | 1 | 3 program wait states inserted |

Area 7 Wait Control

| | | |
|---|---|---|
| 0 | 0 | Program wait not inserted |
| | 1 | 1 program wait state inserted |
| 1 | 0 | 2 program wait states inserted |
| | 1 | 3 program wait states inserted |

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | W31 | W30 | W21 | W20 | W11 | W10 | W01 | W00 |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Area 0 Wait Control

| | | |
|---|---|---|
| 0 | 0 | Program wait not inserted |
| | 1 | 1 program wait state inserted |
| 1 | 0 | 2 program wait states inserted |
| | 1 | 3 program wait states inserted |

Area 1 Wait Control

| | | |
|---|---|---|
| 0 | 0 | Program wait not inserted |
| | 1 | 1 program wait state inserted |
| 1 | 0 | 2 program wait states inserted |
| | 1 | 3 program wait states inserted |

Area 2 Wait Control

| | | |
|---|---|---|
| 0 | 0 | Program wait not inserted |
| | 1 | 1 program wait state inserted |
| 1 | 0 | 2 program wait states inserted |
| | 1 | 3 program wait states inserted |

Area 3 Wait Control

| | | |
|---|---|---|
| 0 | 0 | Program wait not inserted |
| | 1 | 1 program wait state inserted |
| 1 | 0 | 2 program wait states inserted |
| | 1 | 3 program wait states inserted |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ICIS1 | ICIS0 | BRSTRM | BRSTS1 | BRSTS0 | — | — | — |
| Initial value | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Reserved
Only 0 should be written to these bits

Burst Cycle Select 0

| | |
|---|---|
| 0 | Max. 4 words in burst access |
| 1 | Max. 8 words in burst access |

Burst Cycle Select 1

| | |
|---|---|
| 0 | Burst cycle comprises 1 state |
| 1 | Burst cycle comprises 2 states |

Area 0 Burst ROM Enable

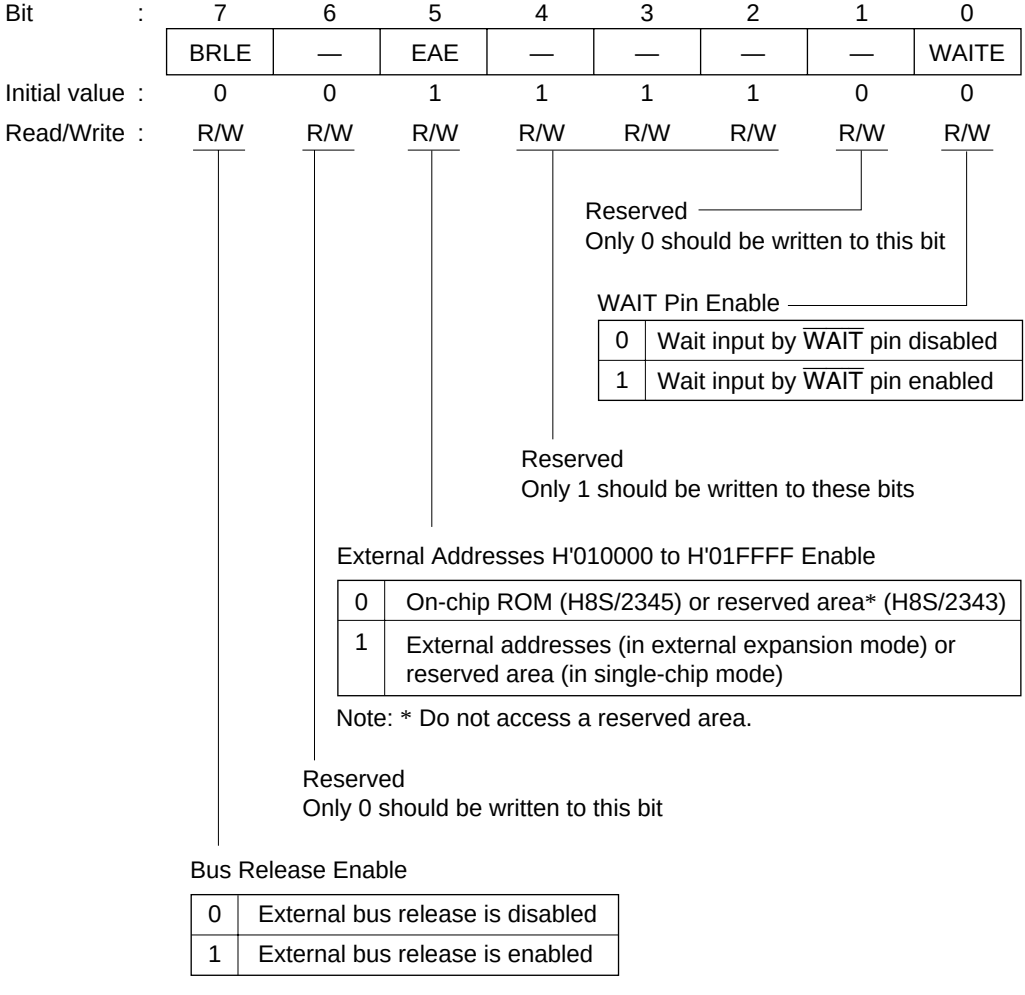
| | |
|---|---|
| 0 | Area 0 is basic bus interface |
| 1 | Area 0 is burst ROM interface |

Idle Cycle Insert 0

| | |
|---|---|
| 0 | Idle cycle not inserted in case of successive external read and external write cycles |
| 1 | Idle cycle inserted in case of successive external read and external write cycles |

Idle Cycle Insert 1

| | |
|---|---|
| 0 | Idle cycle not inserted in case of successive external read cycles in different areas |
| 1 | Idle cycle inserted in case of successive external read cycles in different areas |


Bit : 7 6 5 4 3 2 1 0
BRLE — EAE — — — — WAITE
Initial value : 0 0 1 1 1 1 0 0
Read/Write : R/W R/W R/W R/W R/W R/W R/W R/W
Reserved
Only 0 should be written to this bit
WAIT Pin Enable
0 Wait input by WAIT pin disabled
1 Wait input by WAIT pin enabled
Reserved
Only 1 should be written to these bits
External Addresses H'010000 to H'01FFFF Enable
0 On-chip ROM (H8S/2345) or reserved area* (H8S/2343)
1 External addresses (in external expansion mode) or reserved area (in single-chip mode)
Note: * Do not access a reserved area.
Reserved
Only 0 should be written to this bit
Bus Release Enable
0 External bus release is disabled
1 External bus release is enabled

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | — | RAMS | RAM1 | RAM0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | — | — | — | — | — | R/W | R/W | R/W |

RAM Select, Flash Memory Area Select

| RAMS | RAM1 | RAM0 | RAM Area |
|---|---|---|---|
| 0 | * | * | H'FFEC00–H'FFEFFF |
| 1 | 0 | 0 | H'000000–H'0003FF |
| | | 1 | H'000400–H'0007FF |
| | 1 | 0 | H'000800–H'000BFF |
| | | 1 | H'000C00–H'000FFF |

*: Don't care

ISCRH

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| | | IRQ7SCB | IRQ7SCA | IRQ6SCB | IRQ6SCA | IRQ5SCB | IRQ5SCA | IRQ4SCB | IRQ4SCA |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

IRQ7 to IRQ4 Sense Control

ISCRL

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | IRQ3SCB | IRQ3SCA | IRQ2SCB | IRQ2SCA | IRQ1SCB | IRQ1SCA | IRQ0SCB | IRQ0SCA |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

IRQ3 to IRQ0 Sense Control

| IRQnSCB | IRQnSCA | Interrupt Request Generation |
|---|---|---|
| 0 | 0 | $\overline{\text{IRQn}}$ input low level |
| | 1 | Falling edge of $\overline{\text{IRQn}}$ input |
| 1 | 0 | Rising edge of $\overline{\text{IRQn}}$ input |
| | 1 | Both falling and rising edges of $\overline{\text{IRQn}}$ input |

(n = 7 to 0)

**IER—IRQ Enable Register** **H'FF2E** **Interrupt Controller**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IRQ7E | IRQ6E | IRQ5E | IRQ4E | IRQ3E | IRQ2E | IRQ1E | IRQ0E |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

IRQn Enable

| | |
|---|---|
| 0 | IRQn interrupt disabled |
| 1 | IRQn interrupt enabled |

(n = 7 to 0)

**ISR—IRQ Status Register** **H'FF2F** **Interrupt Controller**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IRQ7F | IRQ6F | IRQ5F | IRQ4F | IRQ3F | IRQ2F | IRQ1F | IRQ0F |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* |

Indicate the status of IRQ7 to IRQ0 interrupt requests

Note: * Can only be written with 0 for flag clearing.

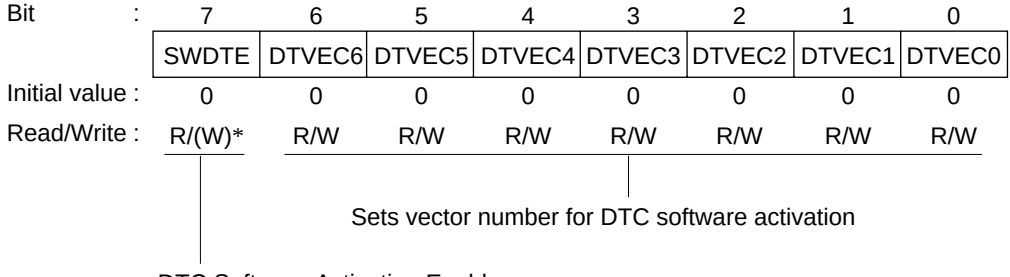
| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | DTCE7 | DTCE6 | DTCE5 | DTCE4 | DTCE3 | DTCE2 | DTCE1 | DTCE0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

DTC Activation Enable

| | |
|---|---|
| 0 | DTC activation by this interrupt is disabled<br>[Clearing conditions]<br>• When the DISEL bit is 1 and data transfer has ended<br>•When the specified number of transfers have ended |
| 1 | DTC activation by this interrupt is enabled<br>[Holding condition]<br>When the DISEL bit is 0 and the specified number of transfers have not ended |

### Correspondence between Interrupt Sources and DTCER

| | Bits | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Register** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| DTCERA | IRQ0 | IRQ1 | IRQ2 | IRQ3 | IRQ4 | IRQ5 | IRQ6 | IRQ7 |
| DTCERB | — | ADI | TGI0A | TGI0B | TGI0C | TGI0D | TGI1A | TGI1B |
| DTCERC | TGI2A | TGI2B | TGI3A | TGI3B | TGI3C | TGI3D | TGI4A | TGI4B |
| DTCERD | — | — | TGI5A | TGI5B | CMIA0 | CMIB0 | CMIA1 | CMIB1 |
| DTCERE | — | — | — | — | RXI0 | TXI0 | RXI1 | TXI1 |




DTC Software Activation Enable

| | |
|---|---|
| 0 | DTC software activation is disabled<br>[Clearing condition]<br>When the DISEL bit is 0 and the specified number of transfers have not ended |
| 1 | DTC software activation is enabled<br>[Holding conditions]<br>• When the DISEL bit is 1 and data transfer has ended<br>• When the specified number of transfers have ended<br>• During data transfer due to software activation |

Note: * A value of 1 can always be written to the SWDTE bit, but 0 can only be written after 1 is read.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SSBY | STS2 | STS1 | STS0 | OPE | — | — | — |
| Initial value | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | — | — | R/W |

Reserved
Only 0 should be written to this bit

Output Port Enable

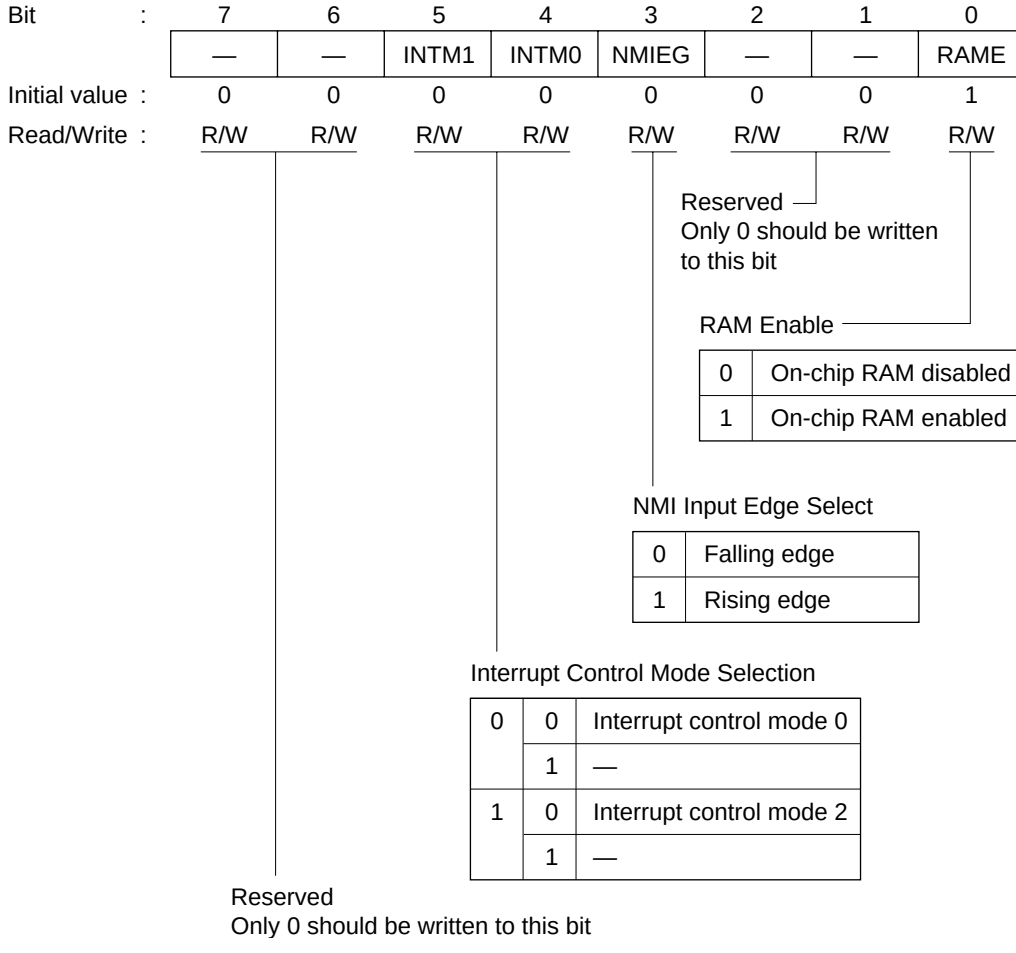
| | |
|---|---|
| 0 | In software standby mode, address bus and bus control signals are high-impedance |
| 1 | In software standby mode, address bus and bus control signals retain output state |

Standby Timer Select

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | Standby time = 8192 states |
| | | 1 | Standby time = 16384 states |
| | 1 | 0 | Standby time = 32768 states |
| | | 1 | Standby time = 65536 states |
| 1 | 0 | 0 | Standby time = 131072 states |
| | | 1 | Standby time = 262144 states |
| | 1 | 0 | Reserved |
| | | 1 | Standby time = 16 states |

Software Standby

| | |
|---|---|
| 0 | Transition to sleep mode after execution of SLEEP instruction |
| 1 | Transition to software standby mode after execution of SLEEP instruction |


Bit : 7 6 5 4 3 2 1 0
— — INTM1 INTM0 NMIEG — — RAME
Initial value : 0 0 0 0 0 0 0 1
Read/Write : R/W R/W R/W R/W R/W R/W R/W R/W
Reserved
Only 0 should be written to this bit
RAM Enable
0 On-chip RAM disabled
1 On-chip RAM enabled
NMI Input Edge Select
0 Falling edge
1 Rising edge
Interrupt Control Mode Selection
0 0 Interrupt control mode 0
1 —
1 0 Interrupt control mode 2
1 —
Reserved
Only 0 should be written to this bit

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PSTOP | — | — | — | — | SCK2 | SCK1 | SCK0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | — | — | — | R/W | R/W | R/W |

Bus Master Clock Select

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | Bus master is in high-speed mode |
| | | 1 | Medium-speed clock is ø/2 |
| | 1 | 0 | Medium-speed clock is ø/4 |
| | | 1 | Medium-speed clock is ø/8 |
| 1 | 0 | 0 | Medium-speed clock is ø/16 |
| | | 1 | Medium-speed clock is ø/32 |
| | 1 | — | — |

ø Clock Output Control

| PSTOP | Normal Operation | Sleep Mode | Software Standby Mode | Hardware Standby Mode |
|---|---|---|---|---|
| 0 | ø output | ø output | Fixed high | High impedance |
| 1 | Fixed high | Fixed high | Fixed high | High impedance |

**MDCR—Mode Control Register H'FF3B MCU**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | MDS2 | MDS1 | MDS0 |
| Initial value | 1 | 0 | 0 | 0 | 0 | —* | —* | —* |
| Read/Write | — | — | — | — | — | R | R | R |

Current mode pin operating mode

Note: * Determined by pins $MD_2$ to $MD_0$

**MSTPCRH — Module Stop Control Register H H'FF3C Power-Down State**
**MSTPCRL — Module Stop Control Register L H'FF3D Power-Down State**

| | MSTPCRH | | | | | | | | MSTPCRL | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Specifies module stop mode

| | |
|---|---|
| 0 | Module stop mode cleared |
| 1 | Module stop mode set |

**SYSCR2 — System Control Register 2 H'FF42 MCU**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | FLSHE | — | — | — |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | — | — | R/W | — | — | — |

Flash Memory Control Register Enable

| | |
|---|---|
| 0 | Flash memory control register is not selected |
| 1 | Flash memory control register is selected |

**Reserved Register** **H'FF44**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | — | — | — |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | R/W | — | — | — | — | — |

Reserved
Only 0 should be written to these bits

**Reserved Register** **H'FF45**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | — | — | — |
| Initial value | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Reserved

**PORT1—Port 1 Register** **H'FF50** **Port 1**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 |
| Initial value | —* | —* | —* | —* | —* | —* | —* | —* |
| Read/Write | R | R | R | R | R | R | R | R |

State of port 1 pins

Note: * Determined by the state of pins $P1_7$ to $P1_0$.

**PORT2—Port 2 Register** **H'FF51** **Port 2**

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P27 | P26 | P25 | P24 | P23 | P22 | P21 | P20 |
| Initial value : | —* | —* | —* | —* | —* | —* | —* | —* |
| Read/Write : | R | R | R | R | R | R | R | R |

State of port 2 pins

Note: * Determined by the state of pins $P2_7$ to $P2_0$.

**PORT3—Port 3 Register** **H'FF52** **Port 3**

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | P35 | P34 | P33 | P32 | P31 | P30 |
| Initial value : | Undefined | Undefined | —* | —* | —* | —* | —* | —* |
| Read/Write : | — | — | R | R | R | R | R | R |

State of port 3 pins

Note: * Determined by the state of pins $P3_5$ to $P3_0$.

**PORT4—Port 4 Register** **H'FF53** **Port 4**

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P47 | P46 | P45 | P44 | P43 | P42 | P41 | P40 |
| Initial value : | —* | —* | —* | —* | —* | —* | —* | —* |
| Read/Write : | R | R | R | R | R | R | R | R |

State of port 4 pins

Note: * Determined by the state of pins $P4_7$ to $P4_0$.

**PORTA—Port A Register** **H'FF59** **Port A**

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | PA3 | PA2 | PA1 | PA0 |
| Initial value : | Undefined | Undefined | Undefined | Undefined | —* | —* | —* | —* |
| Read/Write : | — | — | — | — | R | R | R | R |

State of port A pins

Note: * Determined by the state of pins $PA_3$ to $PA_0$.

**PORTB—Port B Register** **H'FF5A** **Port B**

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PB7 | PB6 | PB5 | PB4 | PB3 | PB2 | PB1 | PB0 |
| Initial value : | —* | —* | —* | —* | —* | —* | —* | —* |
| Read/Write : | R | R | R | R | R | R | R | R |

State of port B pins

Note: * Determined by the state of pins $PB_7$ to $PB_0$.

**PORTC—Port C Register** **H'FF5B** **Port C**

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PC7 | PC6 | PC5 | PC4 | PC3 | PC2 | PC1 | PC0 |
| Initial value : | —* | —* | —* | —* | —* | —* | —* | —* |
| Read/Write : | R | R | R | R | R | R | R | R |

State of port C pins

Note: * Determined by the state of pins $PC_7$ to $PC_0$.

**PORTD—Port D Register** **H'FF5C** **Port D**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PD7 | PD6 | PD5 | PD4 | PD3 | PD2 | PD1 | PD0 |
| Initial value | —* | —* | —* | —* | —* | —* | —* | —* |
| Read/Write | R | R | R | R | R | R | R | R |

State of port D pins

Note: * Determined by the state of pins $PD_7$ to $PD_0$.

**PORTE—Port E Register** **H'FF5D** **Port E**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PE7 | PE6 | PE5 | PE4 | PE3 | PE2 | PE1 | PE0 |
| Initial value | —* | —* | —* | —* | —* | —* | —* | —* |
| Read/Write | R | R | R | R | R | R | R | R |

State of port E pins

Note: * Determined by the state of pins $PE_7$ to $PE_0$.

**PORTF—Port F Register** **H'FF5E** **Port F**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PF7 | PF6 | PF5 | PF4 | PF3 | PF2 | PF1 | PF0 |
| Initial value | —* | —* | —* | —* | —* | —* | —* | —* |
| Read/Write | R | R | R | R | R | R | R | R |

State of port F pins

Note: * Determined by the state of pins $PF_7$ to $PF_0$.

**PORTG—Port G Register** **H'FF5F** **Port G**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | PG4 | PG3 | PG2 | PG1 | PG0 |
| Initial value | Undefined | Undefined | Undefined | —* | —* | —* | —* | —* |
| Read/Write | — | — | — | R | R | R | R | R |

State of port G pins

Note: * Determined by the state of pins $PG_4$ to $PG_0$.

**P1DR—Port 1 Data Register** **H'FF60** **Port 1**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P17DR | P16DR | P15DR | P14DR | P13DR | P12DR | P11DR | P10DR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Stores output data for port 1 pins ($P1_7$ to $P1_0$)

**P2DR—Port 2 Data Register** **H'FF61** **Port 2**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P27DR | P26DR | P25DR | P24DR | P23DR | P22DR | P21DR | P20DR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Stores output data for port 2 pins ($P2_7$ to $P2_0$)

**P3DR—Port 3 Data Register** **H'FF62** **Port 3**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | P35DR | P34DR | P33DR | P32DR | P31DR | P30DR |
| Initial value | Undefined | Undefined | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

Stores output data for port 3 pins ($P3_5$ to $P3_0$)

**PADR—Port A Data Register** **H'FF69** **Port A**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | PA3DR | PA2DR | PA1DR | PA0DR |
| Initial value | Undefined | Undefined | Undefined | Undefined | 0 | 0 | 0 | 0 |
| Read/Write | — | — | — | — | R/W | R/W | R/W | R/W |

Stores output data for port A pins ($PA_3$ to $PA_0$)

**PBDR—Port B Data Register** **H'FF6A** **Port B**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PB7DR | PB6DR | PB5DR | PB4DR | PB3DR | PB2DR | PB1DR | PB0DR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Stores output data for port B pins ($PB_7$ to $PB_0$)

**PCDR—Port C Data Register** **H'FF6B** **Port C**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PC7DR | PC6DR | PC5DR | PC4DR | PC3DR | PC2DR | PC1DR | PC0DR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Stores output data for port C pins ($PC_7$ to $PC_0$)

**PDDR—Port D Data Register** **H'FF6C** **Port D**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PD7DR | PD6DR | PD5DR | PD4DR | PD3DR | PD2DR | PD1DR | PD0DR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Stores output data for port D pins ($PD_7$ to $PD_0$)

**PEDR—Port E Data Register** **H'FF6D** **Port E**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PE7DR | PE6DR | PE5DR | PE4DR | PE3DR | PE2DR | PE1DR | PE0DR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Stores output data for port E pins ($PE_7$ to $PE_0$)

**PFDR—Port F Data Register** **H'FF6E** **Port F**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PF7DR | PF6DR | PF5DR | PF4DR | PF3DR | PF2DR | PF1DR | PF0DR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Stores output data for port F pins ($PF_7$ to $PF_0$)

**PGDR—Port G Data Register** **H'FF6F** **Port G**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | PG4DR | PG3DR | PG2DR | PG1DR | PG0DR |
| Initial value | Undefined | Undefined | Undefined | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | — | R/W | R/W | R/W | R/W | R/W |

Stores output data for port G pins ($PG_4$ to $PG_0$)

**PAPCR—Port A MOS Pull-Up Control Register** **H'FF70** **Port A**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | PA3PCR | PA2PCR | PA1PCR | PA0PCR |
| Initial value | Undefined | Undefined | Undefined | Undefined | 0 | 0 | 0 | 0 |
| Read/Write | — | — | — | — | R/W | R/W | R/W | R/W |

Controls the MOS input pull-up function incorporated into port A on a bit-by-bit basis

**PBPCR—Port B MOS Pull-Up Control Register** **H'FF71** **Port B**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PB7PCR | PB6PCR | PB5PCR | PB4PCR | PB3PCR | PB2PCR | PB1PCR | PB0PCR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Controls the MOS input pull-up function incorporated into port B on a bit-by-bit basis

**PCPCR—Port C MOS Pull-Up Control Register** **H'FF72** **Port C**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PC7PCR | PC6PCR | PC5PCR | PC4PCR | PC3PCR | PC2PCR | PC1PCR | PC0PCR |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Controls the MOS input pull-up function incorporated into port C on a bit-by-bit basis

**PDPCR—Port D MOS Pull-Up Control Register H'FF73 Port D**

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PD7PCR | PD6PCR | PD5PCR | PD4PCR | PD3PCR | PD2PCR | PD1PCR | PD0PCR |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Controls the MOS input pull-up function incorporated into port D on a bit-by-bit basis

**PEPCR—Port E MOS Pull-Up Control Register H'FF74 Port E**

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PE7PCR | PE6PCR | PE5PCR | PE4PCR | PE3PCR | PE2PCR | PE1PCR | PE0PCR |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Controls the MOS input pull-up function incorporated into port E on a bit-by-bit basis

**P3ODR—Port 3 Open Drain Control Register H'FF76 Port 3**

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | P35ODR | P34ODR | P33ODR | P32ODR | P31ODR | P30ODR |
| Initial value : | Undefined | Undefined | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

Controls the PMOS on/off status for each port 3 pin ($P3_5$ to $P3_0$)

**PAODR—Port A Open Drain Control Register H'FF77 Port A**

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | PA3ODR | PA2ODR | PA1ODR | PA0ODR |
| Initial value : | Undefined | Undefined | Undefined | Undefined | 0 | 0 | 0 | 0 |
| Read/Write : | — | — | — | — | R/W | R/W | R/W | R/W |

Controls the PMOS on/off status for each port A pin ($PA_3$ to $PA_0$)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | C/$\overline{A}$ | CHR | PE | O/$\overline{E}$ | STOP | MP | CKS1 | CKS0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Clock Select

| | | |
|---|---|---|
| 0 | 0 | ø clock |
| | 1 | ø/4 clock |
| 1 | 0 | ø/16 clock |
| | 1 | ø/64 clock |

Multiprocessor Mode

| | |
|---|---|
| 0 | Multiprocessor function disabled |
| 1 | Multiprocessor format selected |

Stop Bit Length

| | |
|---|---|
| 0 | 1 stop bit |
| 1 | 2 stop bits |

Parity Mode

| | |
|---|---|
| 0 | Even parity |
| 1 | Odd parity |

Parity Enable

| | |
|---|---|
| 0 | Parity bit addition and checking disabled |
| 1 | Parity bit addition and checking enabled |

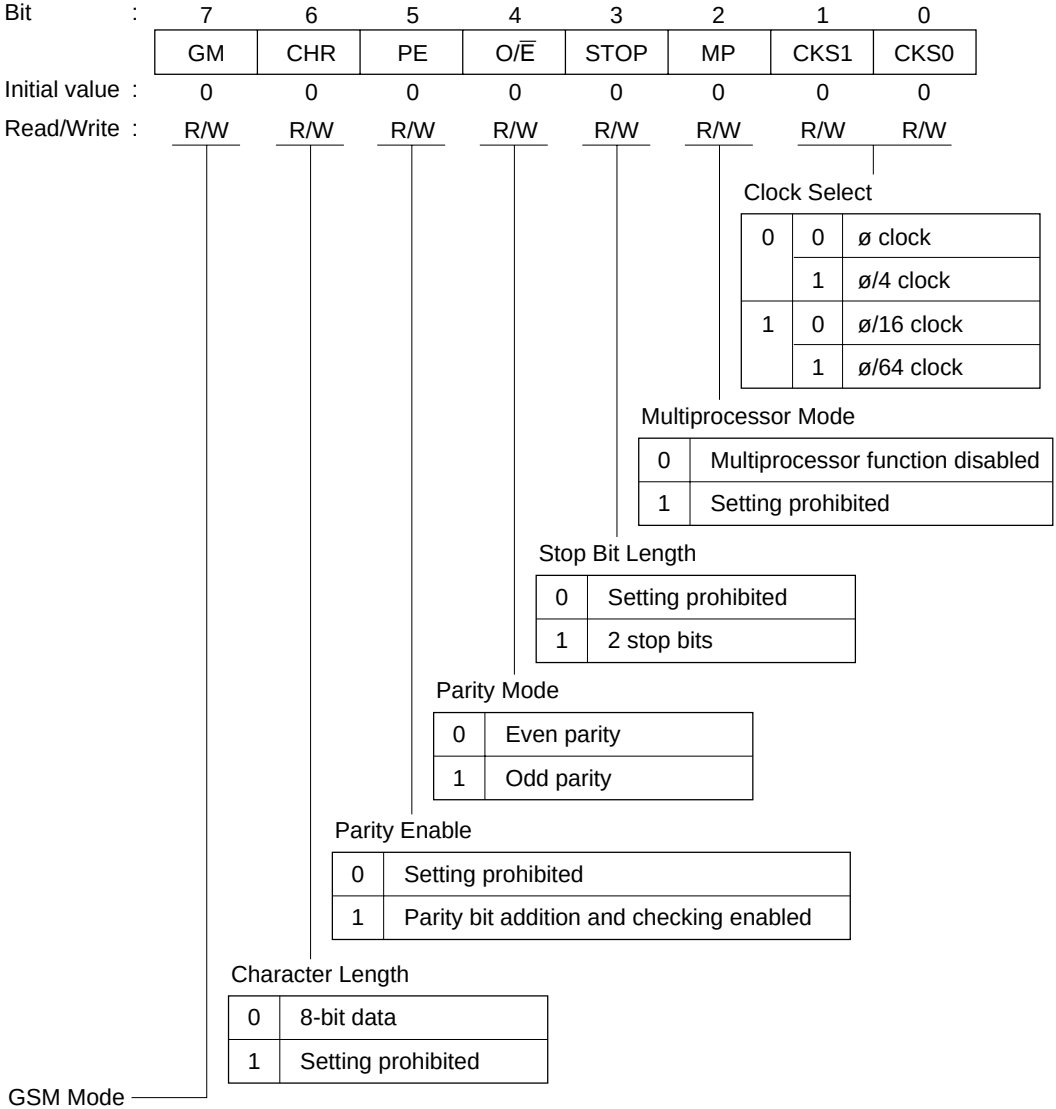
Character Length

| | |
|---|---|
| 0 | 8-bit data |
| 1 | 7-bit data* |

Note: * When 7-bit data is selected, the MSB (bit 7) of TDR is not transmitted.

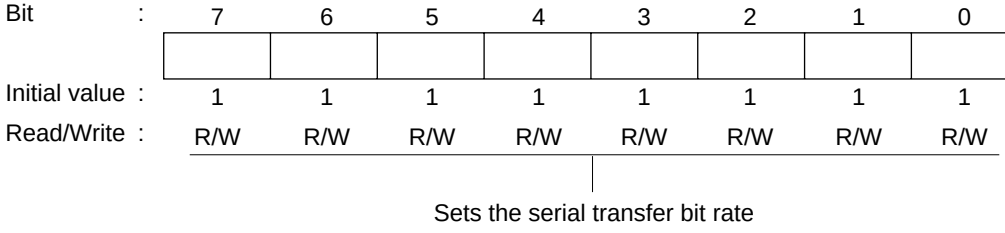
Asynchronous Mode/Synchronous Mode Select

| | |
|---|---|
| 0 | Asynchronous mode |
| 1 | Synchronous mode |




| | |
|---|---|
| 0 | Normal smart card interface mode operation<br>• TEND flag generated 12.5 etu after beginning of start bit<br>• Clock output on/off control only |
| 1 | GSM mode smart card interface mode operation<br>• TEND flag generated 11.0 etu after beginning of start bit<br>• Fixed high/low-level control possible (set in SCR) in addition to clock output on/off control |

Note: etu (Elementary Time Unit): Interval for transfer of one bit




Note: See section 12.2.8, Bit Rate Register (BRR), for details.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Clock Enable

| | | | |
|---|---|---|---|
| 0 | 0 | Asynchronous mode | Internal clock/SCK pin functions as I/O port |
| | | Synchronous mode | Internal clock/SCK pin functions as serial clock output |
| | 1 | Asynchronous mode | Internal clock/SCK pin functions as clock output*1 |
| | | Synchronous mode | Internal clock/SCK pin functions as serial clock output |
| 1 | 0 | Asynchronous mode | External clock/SCK pin functions as clock input*2 |
| | | Synchronous mode | External clock/SCK pin functions as serial clock input |
| | 1 | Asynchronous mode | External clock/SCK pin functions as clock input*2 |
| | | Synchronous mode | External clock/SCK pin functions as serial clock input |

Notes: 1. Outputs a clock of the same frequency as the bit rate.
2. Inputs a clock with a frequency 16 times the bit rate.

Transmit End Interrupt Enable

| | |
|---|---|
| 0 | Transmit end interrupt (TEI) request disabled |
| 1 | Transmit end interrupt (TEI) request enabled |

Multiprocessor Interrupt Enable

| | |
|---|---|
| 0 | Multiprocessor interrupts disabled<br>[Clearing conditions]<br>• When the MPIE bit is cleared to 0<br>• When MPB= 1 data is received |
| 1 | Multiprocessor interrupts enabled<br>Receive interrupt (RXI) requests, receive error interrupt (ERI) requests, and setting of the RDRF, FER, and ORER flags in SSR are disabled until data with the multiprocessor bit set to 1 is received |

Receive Enable

| | |
|---|---|
| 0 | Reception disabled |
| 1 | Reception enabled |

Transmit Enable

| | |
|---|---|
| 0 | Transmission disabled |
| 1 | Transmission enabled |

Receive Interrupt Enable

| | |
|---|---|
| 0 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request disabled |
| 1 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request enabled |

Transmit Interrupt Enable

| | |
|---|---|
| 0 | Transmit data empty interrupt (TXI) requests disabled |
| 1 | Transmit data empty interrupt (TXI) requests enabled |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Clock Enable**

| SMCR | SMR | SCR setting | | SCK pin function |
|---|---|---|---|---|
| SMIF | C/$\overline{A}$,GM | CKE1 | CKE0 | |
| 0 | See SCI specification | | | |
| 1 | 0 | 0 | 0 | Operates as port input pin |
| 1 | 0 | 0 | 1 | Clock output as SCK output pin |
| 1 | 1 | 0 | 0 | Fixed-low output as SCK output pin |
| 1 | 1 | 0 | 1 | Clock output as SCK output pin |
| 1 | 1 | 1 | 0 | Fixed-high output as SCK output pin |
| 1 | 1 | 1 | 1 | Clock output as SCK output pin |

**Transmit End Interrupt Enable**

| | |
|---|---|
| 0 | Transmit end interrupt (TEI) request disabled |
| 1 | Transmit end interrupt (TEI) request enabled |

**Multiprocessor Interrupt Enable**

| | |
|---|---|
| 0 | Multiprocessor interrupts disabled<br>[Clearing conditions]<br>• When the MPIE bit is cleared to 0<br>• When MPB= 1 data is received |
| 1 | Multiprocessor interrupts enabled<br>Receive interrupt (RXI) requests, receive error interrupt (ERI) requests, and setting of the RDRF, FER, and ORER flags in SSR are disabled until data with the multiprocessor bit set to 1 is received |

**Receive Enable**

| | |
|---|---|
| 0 | Reception disabled |
| 1 | Reception enabled |

**Transmit Enable**

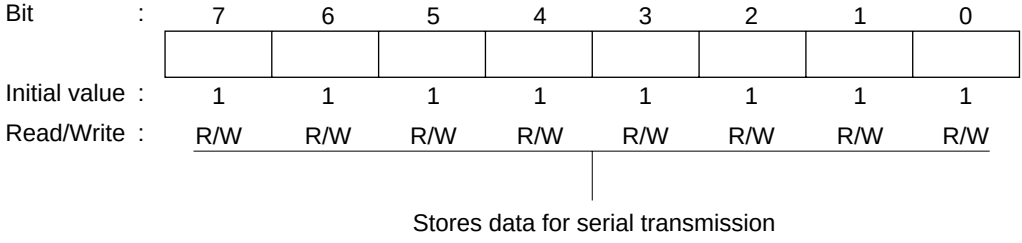
| | |
|---|---|
| 0 | Transmission disabled |
| 1 | Transmission enabled |

**Receive Interrupt Enable**

| | |
|---|---|
| 0 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request disabled |
| 1 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request enabled |

**Transmit Interrupt Enable**

| | |
|---|---|
| 0 | Transmit data empty interrupt (TXI) requests disabled |
| 1 | Transmit data empty interrupt (TXI) requests enabled |


Bit :
7
6
5
4
3
2
1
0
Initial value :
1
1
1
1
1
1
1
1
Read/Write :
R/W
R/W
R/W
R/W
R/W
R/W
R/W
R/W
Stores data for serial transmission

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TDRE | RDRF | ORER | FER | PER | TEND | MPB | MPBT |
| Initial value | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Read/Write | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R | R | R/W |

**Multiprocessor Bit Transfer**

| | |
|---|---|
| 0 | Data with a 0 multiprocessor bit is transmitted |
| 1 | Data with a 1 multiprocessor bit is transmitted |

**Multiprocessor Bit**

| | |
|---|---|
| 0 | [Clearing condition]<br>When data with a 0 multiprocessor bit is received |
| 1 | [Setting condition]<br>When data with a 1 multiprocessor bit is received |

**Transmit End**

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DTC is activated by a TXI interrupt and writes data to TDR |
| 1 | [Setting conditions]<br>• When the TE bit in SCR is 0<br>• When TDRE = 1 at transmission of the last bit of a 1-byte serial transmit character |

**Parity Error**

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to PER after reading PER = 1 |
| 1 | [Setting condition]<br>When, in reception, the number of 1 bits in the receive data plus the parity bit does not match the parity setting (even or odd) specified by the O/$\overline{E}$ bit in SMR |

**Framing Error**

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to FER after reading FER = 1 |
| 1 | [Setting condition]<br>When the SCI checks whether the stop bit at the end of the receive data when reception ends, and the stop bit is 0 |

**Overrun Error**

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to ORER after reading ORER = 1 |
| 1 | [Setting condition]<br>When the next serial reception is completed while RDRF = 1 |

**Receive Data Register Full**

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to RDRF after reading RDRF = 1<br>• When the DTC is activated by an RXI interrupt and reads data from RDR |
| 1 | [Setting condition]<br>When serial reception ends normally and receive data is transferred from RSR to RDR |

**Transmit Data Register Empty**

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DTC is activated by a TXI interrupt and writes data to TDR |
| 1 | [Setting condition]<br>• When the TE bit in SCR is 0<br>• When data is transferred from TDR to TSR and data can be written to TDR |

Note: * Can only be written with 0 for flag clearing.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TDRE | RDRF | ORER | ERS | PER | TEND | MPB | MPBT |
| Initial value | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Read/Write | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R | R | R/W |

Multiprocessor Bit Transfer

| | |
|---|---|
| 0 | Data with a 0 multiprocessor bit is transmitted |
| 1 | Data with a 1 multiprocessor bit is transmitted |

Multiprocessor Bit

| | |
|---|---|
| 0 | [Clearing condition]<br>When data with a 0 multiprocessor bit is received |
| 1 | [Setting condition]<br>When data with a 1 multiprocessor bit is received |

Transmit End

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DTC is activated by a TXI interrupt and writes data to TDR |
| 1 | [Setting conditions]<br>• On reset, or in standby mode or module stop mode<br>• When the TE bit in SCR is 0 and the ERS bit is 0<br>• When TDRE = 1 and ERS = 0 (normal transmission) 2.5 etu after a 1-byte serial character is transmitted when GM = 0<br>• When TDRE = 1 and ERS = 0 (normal transmission) 1.0 etu after a 1-byte serial character is transmitted when GM = 1 |

Note: etu: Elementary Time Unit (the time taken to transmit one bit)

Parity Error

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to PER after reading PER = 1 |
| 1 | [Setting condition]<br>When, in reception, the number of 1 bits in the receive data plus the parity bit does not match the parity setting (even or odd) specified by the O/$\overline{E}$ bit in SMR |

Error Signal Status

| | |
|---|---|
| 0 | [Clearing condition]<br>• On reset, or in standby mode or module stop mode<br>• When 0 is written to ERS after reading ERS = 1 |
| 1 | [Setting condition]<br>When the error signal is sampled at the low level |

Note: Clearing the TE bit in SCR to 0 does not affect the ERS flag, which retains its prior state.

Overrun Error

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to ORER after reading ORER = 1 |
| 1 | [Setting condition]<br>When the next serial reception is completed while RDRF = 1 |

Receive Data Register Full

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to RDRF after reading RDRF = 1<br>• When the DTC is activated by an RXI interrupt and reads data from RDR |
| 1 | [Setting condition]<br>When serial reception ends normally and receive data is transferred from RSR to RDR |

Transmit Data Register Empty

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DTC is activated by a TXI interrupt and writes data to TDR |
| 1 | [Setting condition]<br>• When the TE bit in SCR is 0<br>• When data is transferred from TDR to TSR and data can be written to TDR |

Note: * Can only be written with 0 for flag clearing.

**RDR0—Receive Data Register 0** **H'FF7D** **SCI0, Smart Card Interface 0**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R | R | R | R | R | R | R | R |

Stores received serial data

**SCMR0—Smart Card Mode Register 0** **H'FF7E** **SCI0, Smart Card Interface 0**

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | SDIR | SINV | — | SMIF |
| Initial value | : | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| Read/Write | : | — | — | — | — | R/W | R/W | — | R/W |

Smart Card Interface Mode Select

| | |
|---|---|
| 0 | Smart Card interface function is disabled |
| 1 | Smart Card interface function is enabled |

Smart Card Data Invert

| | |
|---|---|
| 0 | TDR contents are transmitted as they are<br>Receive data is stored in RDR as it is |
| 1 | TDR contents are inverted before being transmitted<br>Receive data is stored in RDR in inverted form |

Smart Card Data Direction

| | |
|---|---|
| 0 | TDR contents are transmitted LSB-first<br>Receive data is stored in RDR LSB-first |
| 1 | TDR contents are transmitted MSB-first<br>Receive data is stored in RDR MSB-first |

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | C/$\overline{A}$ | CHR | PE | O/$\overline{E}$ | STOP | MP | CKS1 | CKS0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Clock Select

| | | |
|---|---|---|
| 0 | 0 | ø clock |
| | 1 | ø/4 clock |
| 1 | 0 | ø/16 clock |
| | 1 | ø/64 clock |

Multiprocessor Mode

| | |
|---|---|
| 0 | Multiprocessor function disabled |
| 1 | Multiprocessor format selected |

Stop Bit Length

| | |
|---|---|
| 0 | 1 stop bit |
| 1 | 2 stop bits |

Parity Mode

| | |
|---|---|
| 0 | Even parity |
| 1 | Odd parity |

Parity Enable

| | |
|---|---|
| 0 | Parity bit addition and checking disabled |
| 1 | Parity bit addition and checking enabled |

Character Length

| | |
|---|---|
| 0 | 8-bit data |
| 1 | 7-bit data* |

Note: * When 7-bit data is selected, the MSB (bit 7) of TDR is not transmitted.

Asynchronous Mode/Synchronous Mode Select

| | |
|---|---|
| 0 | Asynchronous mode |
| 1 | Synchronous mode |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | GM | CHR | PE | O/$\overline{E}$ | STOP | MP | CKS1 | CKS0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Clock Select**

| | | |
|---|---|---|
| 0 | 0 | ø clock |
| | 1 | ø/4 clock |
| 1 | 0 | ø/16 clock |
| | 1 | ø/64 clock |

**Multiprocessor Mode**

| | |
|---|---|
| 0 | Multiprocessor function disabled |
| 1 | Setting prohibited |

**Stop Bit Length**

| | |
|---|---|
| 0 | Setting prohibited |
| 1 | 2 stop bits |

**Parity Mode**

| | |
|---|---|
| 0 | Even parity |
| 1 | Odd parity |

**Parity Enable**

| | |
|---|---|
| 0 | Setting prohibited |
| 1 | Parity bit addition and checking enabled |

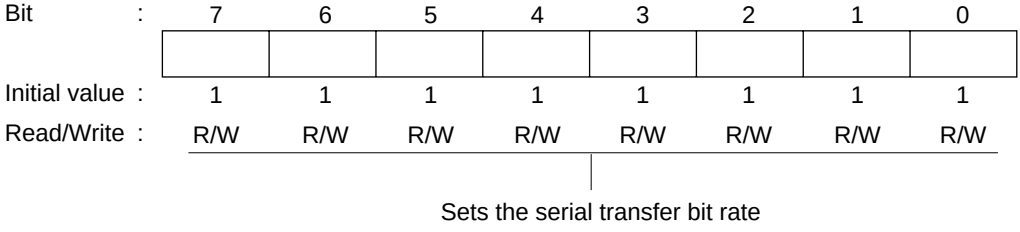
**Character Length**

| | |
|---|---|
| 0 | 8-bit data |
| 1 | Setting prohibited |

**GSM Mode**

| | |
|---|---|
| 0 | Normal smart card interface mode operation<br>• TEND flag generated 12.5 etu after beginning of start bit<br>• Clock output on/off control only |
| 1 | GSM mode smart card interface mode operation<br>• TEND flag generated 11.0 etu after beginning of start bit<br>• Fixed high/low-level control possible (set in SCR) in addition to clock output on/off control |

Note: etu (Elementary Time Unit): Interval for transfer of one bit




Note: See section 12.2.8, Bit Rate Register (BRR), for details.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Clock Enable

| | | | |
|---|---|---|---|
| 0 | 0 | Asynchronous mode | Internal clock/SCK pin functions as I/O port |
| | | Synchronous mode | Internal clock/SCK pin functions as serial clock output |
| | 1 | Asynchronous mode | Internal clock/SCK pin functions as clock output*1 |
| | | Synchronous mode | Internal clock/SCK pin functions as serial clock output |
| 1 | 0 | Asynchronous mode | External clock/SCK pin functions as clock input*2 |
| | | Synchronous mode | External clock/SCK pin functions as serial clock input |
| | 1 | Asynchronous mode | External clock/SCK pin functions as clock input*2 |
| | | Synchronous mode | External clock/SCK pin functions as serial clock input |

Notes: 1. Outputs a clock of the same frequency as the bit rate.
2. Inputs a clock with a frequency 16 times the bit rate.

Transmit End Interrupt Enable

| | |
|---|---|
| 0 | Transmit end interrupt (TEI) request disabled |
| 1 | Transmit end interrupt (TEI) request enabled |

Multiprocessor Interrupt Enable

| | |
|---|---|
| 0 | Multiprocessor interrupts disabled<br>[Clearing conditions]<br>• When the MPIE bit is cleared to 0<br>• When MPB= 1 data is received |
| 1 | Multiprocessor interrupts enabled<br>Receive interrupt (RXI) requests, receive error interrupt (ERI) requests, and setting of the RDRF, FER, and ORER flags in SSR are disabled until data with the multiprocessor bit set to 1 is received |

Receive Enable

| | |
|---|---|
| 0 | Reception disabled |
| 1 | Reception enabled |

Transmit Enable

| | |
|---|---|
| 0 | Transmission disabled |
| 1 | Transmission enabled |

Receive Interrupt Enable

| | |
|---|---|
| 0 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request disabled |
| 1 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request enabled |

Transmit Interrupt Enable

| | |
|---|---|
| 0 | Transmit data empty interrupt (TXI) requests disabled |
| 1 | Transmit data empty interrupt (TXI) requests enabled |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Clock Enable

| SMCR | SMR | SCR setting | | SCK pin function |
|---|---|---|---|---|
| SMIF | C/$\overline{A}$,GM | CKE1 | CKE0 | |
| 0 | See SCI specification | | | |
| 1 | 0 | 0 | 0 | Operates as port input pin |
| 1 | 0 | 0 | 1 | Clock output as SCK output pin |
| 1 | 1 | 0 | 0 | Fixed-low output as SCK output pin |
| 1 | 1 | 0 | 1 | Clock output as SCK output pin |
| 1 | 1 | 1 | 0 | Fixed-high output as SCK output pin |
| 1 | 1 | 1 | 1 | Clock output as SCK output pin |

Transmit End Interrupt Enable

| | |
|---|---|
| 0 | Transmit end interrupt (TEI) request disabled |
| 1 | Transmit end interrupt (TEI) request enabled |

Multiprocessor Interrupt Enable

| | |
|---|---|
| 0 | Multiprocessor interrupts disabled<br>[Clearing conditions]<br>• When the MPIE bit is cleared to 0<br>• When MPB= 1 data is received |
| 1 | Multiprocessor interrupts enabled<br>Receive interrupt (RXI) requests, receive error interrupt (ERI) requests, and setting of the RDRF, FER, and ORER flags in SSR are disabled until data with the multiprocessor bit set to 1 is received |

Receive Enable

| | |
|---|---|
| 0 | Reception disabled |
| 1 | Reception enabled |

Transmit Enable

| | |
|---|---|
| 0 | Transmission disabled |
| 1 | Transmission enabled |

Receive Interrupt Enable

| | |
|---|---|
| 0 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request disabled |
| 1 | Receive data full interrupt (RXI) request and receive error interrupt (ERI) request enabled |

Transmit Interrupt Enable

| | |
|---|---|
| 0 | Transmit data empty interrupt (TXI) requests disabled |
| 1 | Transmit data empty interrupt (TXI) requests enabled |

Bit :
7
6
5
4
3
2
1
0
Initial value :
1
1
1
1
1
1
1
1
Read/Write :
R/W
R/W
R/W
R/W
R/W
R/W
R/W
R/W
Stores data for serial transmission

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TDRE | RDRF | ORER | FER | PER | TEND | MPB | MPBT |
| Initial value | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Read/Write | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R | R | R/W |

Multiprocessor Bit Transfer

| | |
|---|---|
| 0 | Data with a 0 multiprocessor bit is transmitted |
| 1 | Data with a 1 multiprocessor bit is transmitted |

Multiprocessor Bit

| | |
|---|---|
| 0 | [Clearing condition]<br>When data with a 0 multiprocessor bit is received |
| 1 | [Setting condition]<br>When data with a 1 multiprocessor bit is received |

Transmit End

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DTC is activated by a TXI interrupt and writes data to TDR |
| 1 | [Setting condition]<br>• When the TE bit in SCR is 0<br>• When TDRE = 1 at transmission of the last bit of a 1-byte serial transmit character |

Parity Error

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to PER after reading PER = 1 |
| 1 | [Setting condition]<br>When, in reception, the number of 1 bits in the receive data plus the parity bit does not match the parity setting (even or odd) specified by the O/$\overline{E}$ bit in SMR |

Framing Error

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to FER after reading FER = 1 |
| 1 | [Setting condition]<br>When the SCI checks whether the stop bit at the end of the receive data when reception ends, and the stop bit is 0 |

Overrun Error

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to ORER after reading ORER = 1 |
| 1 | [Setting condition]<br>When the next serial reception is completed while RDRF = 1 |

Receive Data Register Full

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to RDRF after reading RDRF = 1<br>• When the DTC is activated by an RXI interrupt and reads data from RDR |
| 1 | [Setting condition]<br>When serial reception ends normally and receive data is transferred from RSR to RDR |

Transmit Data Register Empty

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DTC is activated by a TXI interrupt and writes data to TDR |
| 1 | [Setting condition]<br>• When the TE bit in SCR is 0<br>• When data is transferred from TDR to TSR and data can be written to TDR |

Note: * Can only be written with 0 for flag clearing.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TDRE | RDRF | ORER | ERS | PER | TEND | MPB | MPBT |
| Initial value | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Read/Write | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R | R | R/W |

Multiprocessor Bit Transfer

| | |
|---|---|
| 0 | Data with a 0 multiprocessor bit is transmitted |
| 1 | Data with a 1 multiprocessor bit is transmitted |

Multiprocessor Bit

| | |
|---|---|
| 0 | [Clearing condition]<br>When data with a 0 multiprocessor bit is received |
| 1 | [Setting conditions]<br>When data with a 1 multiprocessor bit is received |

Transmit End

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DTC is activated by a TXI interrupt and writes data to TDR |
| 1 | [Setting conditions]<br>• On reset, or in standby mode or module stop mode<br>• When the TE bit in SCR is 0 and the ERS bit is 0<br>• When TDRE = 1 and ERS = 0 (normal transmission) 2.5 etu after a 1-byte serial character is transmitted when GM = 0<br>• When TDRE = 1 and ERS = 0 (normal transmission) 1.0 etu after a 1-byte serial character is transmitted when GM = 1 |

Note: etu: Elementary Time Unit (the time taken to transmit one bit)

Parity Error

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to PER after reading PER = 1 |
| 1 | [Setting condition]<br>When, in reception, the number of 1 bits in the receive data plus the parity bit does not match the parity setting (even or odd) specified by the O/$\overline{E}$ bit in SMR |

Error Signal Status

| | |
|---|---|
| 0 | [Clearing condition]<br>• On reset, or in standby mode or module stop mode<br>• When 0 is written to ERS after reading ERS =1 |
| 1 | [Setting conditions]<br>When the error signal is sampled at the low level |

Note: Clearing the TE bit in SCR to 0 does not affect the ERS flag, which retains its prior state.

Overrun Error

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to ORER after reading ORER = 1 |
| 1 | [Setting condition]<br>When the next serial reception is completed while RDRF = 1 |

Receive Data Register Full

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to RDRF after reading RDRF = 1<br>• When the DTC is activated by an RXI interrupt and reads data from RDR |
| 1 | [Setting condition]<br>When serial reception ends normally and receive data is transferred from RSR to RDR |

Transmit Data Register Empty

| | |
|---|---|
| 0 | [Clearing condition]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DTC is activated by a TXI interrupt and writes data to TDR |
| 1 | [Setting condition]<br>• When the TE bit in SCR is 0<br>• When data is transferred from TDR to TSR and data can be written to TDR |

Note: * Can only be written with 0 for flag clearing.

**RDR1—Receive Data Register 1** **H'FF85 SCI1, Smart Card Interface 1**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R | R | R | R | R | R | R | R |

Stores received serial data

**SCMR1—Smart Card Mode Register 1** **H'FF86 SCI1, Smart Card Interface 1**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | SDIR | SINV | — | SMIF |
| Initial value | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| Read/Write | — | — | — | — | R/W | R/W | — | R/W |

Smart Card Interface Mode Select

| | |
|---|---|
| 0 | Smart Card interface function is disabled |
| 1 | Smart Card interface function is enabled |

Smart Card Data Invert

| | |
|---|---|
| 0 | TDR contents are transmitted as they are<br>Receive data is stored in RDR as it is |
| 1 | TDR contents are inverted before being transmitted<br>Receive data is stored in RDR in inverted form |

Smart Card Data Direction

| | |
|---|---|
| 0 | TDR contents are transmitted LSB-first<br>Receive data is stored in RDR LSB-first |
| 1 | TDR contents are transmitted MSB-first<br>Receive data is stored in RDR MSB-first |

| | | | | |
|---|---|---|---|---|
| **ADDRAH** | **—** | **A/D Data Register AH** | **H'FF90** | **A/D Converter** |
| **ADDRAL** | **—** | **A/D Data Register AL** | **H'FF91** | **A/D Converter** |
| **ADDRBH** | **—** | **A/D Data Register BH** | **H'FF92** | **A/D Converter** |
| **ADDRBL** | **—** | **A/D Data Register BL** | **H'FF93** | **A/D Converter** |
| **ADDRCH** | **—** | **A/D Data Register CH** | **H'FF94** | **A/D Converter** |
| **ADDRCL** | **—** | **A/D Data Register CL** | **H'FF95** | **A/D Converter** |
| **ADDRDH** | **—** | **A/D Data Register DH** | **H'FF96** | **A/D Converter** |
| **ADDRDL** | **—** | **A/D Data Register DL** | **H'FF97** | **A/D Converter** |

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | AD9 | AD8 | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 | — | — | — | — | — | — |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |

Stores the results of A/D conversion

| Analog Input Channel | | A/D Data Register |
|---|---|---|
| Group 0 | Group 1 | |
| AN0 | AN4 | ADDRA |
| AN1 | AN5 | ADDRB |
| AN2 | AN6 | ADDRC |
| AN3 | AN7 | ADDRD |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADF | ADIE | ADST | SCAN | CKS | CH2 | CH1 | CH0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/(W)* | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Channel Select

| Group select | Channel select | | Single Mode | Scan Mode |
|---|---|---|---|---|
| CH2 | CH1 | CH0 | | |
| 0 | 0 | 0 | AN0 | AN0 |
| | | 1 | AN1 | AN0, AN1 |
| | 1 | 0 | AN2 | AN0 to AN2 |
| | | 1 | AN3 | AN0 to AN3 |
| 1 | 0 | 0 | AN4 | AN4 |
| | | 1 | AN5 | AN4, AN5 |
| | 1 | 0 | AN6 | AN4 to AN6 |
| | | 1 | AN7 | AN4 to AN7 |

Group Select

| | |
|---|---|
| 0 | Conversion time= 266 states (max.) |
| 1 | Conversion time= 134 states (max.) |

Scan Mode

| | |
|---|---|
| 0 | Single mode |
| 1 | Scan mode |

A/D Start

| | |
|---|---|
| 0 | A/D conversion stopped |
| 1 | • Single mode: A/D conversion is started. Cleared to 0 automatically when conversion ends<br>• Scan mode: A/D conversion is started. Conversion continues sequentially on the selected channels until ADST is cleared to 0 by software, a reset, or transition to standby mode or module stop mode |

A/D Interrupt Enable

| | |
|---|---|
| 0 | A/D conversion end interrupt (ADI) request disabled |
| 1 | A/D conversion end interrupt (ADI) request enabled |

A/D End Flag

| | |
|---|---|
| 0 | [Clearing conditions]<br>• When 0 is written to the ADF flag after reading ADF = 1<br>• When the DTC is activated by an ADI interrupt, and ADDR is read |
| 1 | [Setting conditions]<br>• Single mode: When A/D conversion ends<br>• Scan mode: When one round of conversion has been performed on all specified channels |

Note: * Can only be written with 0 for flag clearing.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADF | ADIE | ADST | SCAN | CKS | CH2 | CH1 | CH0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/(W)* | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Channel Select

| Group select | Channel select | | Single Mode | Group Mode |
|---|---|---|---|---|
| CH2 | CH1 | CH0 | | |
| 0 | 0 | 0 | AN0 | AN0 |
| | | 1 | AN1 | AN0, AN1 |
| | 1 | 0 | AN2 | AN0 to AN2 |
| | | 1 | AN3 | AN0 to AN3 |
| 1 | 0 | 0 | AN4 | AN4 |
| | | 1 | AN5 | AN4, AN5 |
| | 1 | 0 | AN6 | AN4 to AN6 |
| | | 1 | AN7 | AN4 to AN7 |

Group Select

| | |
|---|---|
| 0 | Conversion time= 266 states (max.) |
| 1 | Conversion time= 134 states (max.) |

Scan Mode

| | |
|---|---|
| 0 | Single mode |
| 1 | Scan mode |

A/D Start

| | |
|---|---|
| 0 | A/D conversion stopped |
| 1 | • Single mode: A/D conversion is started. Cleared to 0 automatically when conversion ends<br>• Scan mode: A/D conversion is started. Conversion continues sequentially on the selected channels until ADST is cleared to 0 by software, a reset, or transition to standby mode or module stop mode |

A/D Interrupt Enable

| | |
|---|---|
| 0 | A/D conversion end interrupt (ADI) request disabled |
| 1 | A/D conversion end interrupt (ADI) request enabled |

A/D End Flag

| | |
|---|---|
| 0 | [Clearing conditions]<br>• When 0 is written to the ADF flag after reading ADF = 1<br>• When the DTC is activated by an ADI interrupt, and ADDR is read |
| 1 | [Setting conditions]<br>• Single mode: When A/D conversion ends<br>• Scan mode: When one round of conversion has been performed on all specified channels |

Note: * Can only be written with 0 for flag clearing.

**DADR0—D/A Data Register 0** **H'FFA4** **D/A**
**DADR1—D/A Data Register 1** **H'FFA5** **D/A**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DAOE1 | DAOE0 | DAE | — | — | — | — | — |
| Initial value | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | R/W | R/W | R/W | — | — | — | — | — |

D/A Output Enable 0

| | |
|---|---|
| 0 | Analog output DA0 is disabled |
| 1 | Channel 0 D/A conversion is enabled<br>Analog output DA0 is enabled |

D/A Output Enable 1

| | |
|---|---|
| 0 | Analog output DA1 is disabled |
| 1 | Channel 1 D/A conversion is enabled<br>Analog output DA1 is enabled |

D/A Conversion Control

| DAOE1 | DAOE0 | DAE | Description |
|---|---|---|---|
| 0 | 0 | * | Channel 0 and 1 D/A conversion disabled |
| | 1 | 0 | Channel 0 D/A conversion enabled<br>Channel 1 D/A conversion disabled |
| | | 1 | Channel 0 and 1 D/A conversions enabled |
| 1 | 0 | 0 | Channel 0 D/A conversion disabled<br>Channel 1 D/A conversion enabled |
| | | 1 | Channel 0 and 1 D/A conversion enabled |
| | 1 | * | Channel 0 and 1 D/A conversion enabled |

* : Don’t care

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CMIEB | CMIEA | OVIE | CCLR1 | CCLR0 | CKS2 | CKS1 | CKS0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Clock Select

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | Clock input disabled |
| | | 1 | Internal clock: counted at falling edge of ø/8 |
| | 1 | 0 | Internal clock: counted at falling edge of ø/64 |
| | | 1 | Internal clock: counted at falling edge of ø/8192 |
| 1 | 0 | 0 | For channel 0:<br>Count at TCNT1 overflow signal*<br>For channel 1:<br>Count at TCNT0 compare match A* |
| | | 1 | External clock: counted at rising edge |
| | 1 | 0 | External clock: counted at falling edge |
| | | 1 | External clock: counted at both rising and falling edges |

Note: * If the count input of channel 0 is the TCNT1 overflow signal and that of channel 1 is the TCNT0 compare match signal, no incrementing clock is generated. Do not use this setting.

Counter Clear

| | | |
|---|---|---|
| 0 | 0 | Clear is disabled |
| | 1 | Clear by compare match A |
| 1 | 0 | Clear by compare match B |
| | 1 | Clear by rising edge of external reset input |

Timer Overflow Interrupt Enable

| | |
|---|---|
| 0 | OVF interrupt requests (OVI) are disabled |
| 1 | OVF interrupt requests (OVI) are enabled |

Compare Match Interrupt Enable A

| | |
|---|---|
| 0 | CMFA interrupt requests (CMIA) are disabled |
| 1 | CMFA interrupt requests (CMIA) are enabled |

Compare Match Interrupt Enable B

| | |
|---|---|
| 0 | CMFB interrupt requests (CMIB) are disabled |
| 1 | CMFB interrupt requests (CMIB) are enabled |

TCSR0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CMFB | CMFA | OVF | ADTE | OS3 | OS2 | OS1 | OS0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/(W)* | R/(W)* | R/(W)* | R/W | R/W | R/W | R/W | R/W |

TCSR1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CMFB | CMFA | OVF | — | OS3 | OS2 | OS1 | OS0 |
| Initial value | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Read/Write | R/(W)* | R/(W)* | R/(W)* | — | R/W | R/W | R/W | R/W |

Output Select

| | | |
|---|---|---|
| 0 | 0 | No change when compare match A occurs |
| | 1 | 0 is output when compare match A occurs |
| 1 | 0 | 1 is output when compare match A occurs |
| | 1 | Output is inverted when compare match A occurs (toggle output) |

Output Select

| | | |
|---|---|---|
| 0 | 0 | No change when compare match B occurs |
| | 1 | 0 is output when compare match B occurs |
| 1 | 0 | 1 is output when compare match B occurs |
| | 1 | Output is inverted when compare match B occurs (toggle output) |

A/D Trigger Enable (TCSR0 only)

| | |
|---|---|
| 0 | A/D converter start requests by compare match A are disabled |
| 1 | A/D converter start requests by compare match A are enabled |

Timer Overflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>• Cleared by reading OVF when OVF = 1, then writing 0 to OVF |
| 1 | [Setting condition]<br>Set when TCNT overflows (changes from H'FF to H'00) |

Compare Match Flag A

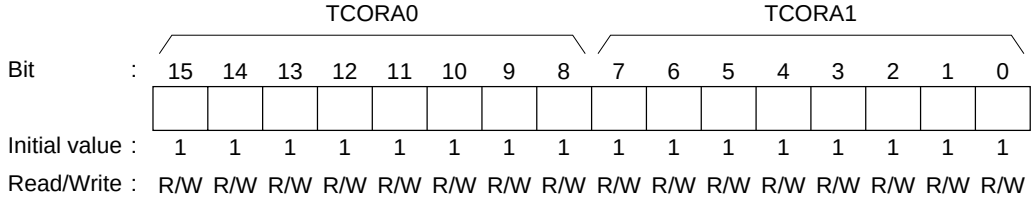
| | |
|---|---|
| 0 | [Clearing condition]<br>• Cleared by reading CMFA when CMFA = 1, then writing 0 to CMFA<br>• When the DTC is activated by a CMIA interrupt, while DISEL bit of MRB in DTC is 0. |
| 1 | [Setting condition]<br>Set when TCNT matches TCORA |

Compare Match Flag B

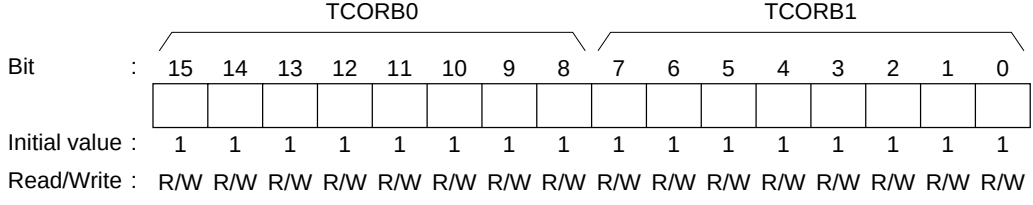
| | |
|---|---|
| 0 | [Clearing condition]<br>• Cleared by reading CMFB when CMFB = 1, then writing 0 to CMFB<br>• When the DTC is activated by a CMIB interrupt, while DISEL bit of MRB in DTC is 0. |
| 1 | [Setting condition]<br>Set when TCNT matches TCORB |

Note: * Only 0 can be written to bits 7 to 5, to clear these flags.

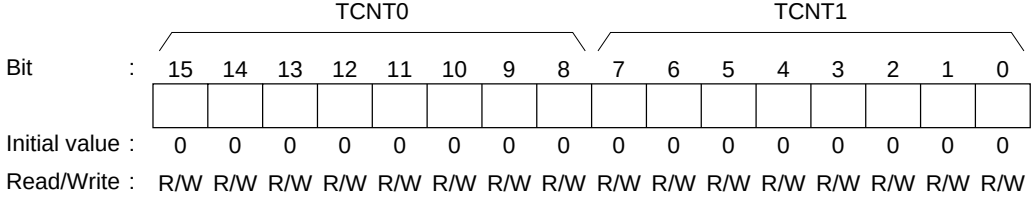
**TCORA0—Time Constant Register A0** **H'FFB4** **8-Bit Timer Channel 0**
**TCORA1—Time Constant Register A1** **H'FFB5** **8-Bit Timer Channel 1**




**TCORB0—Time Constant Register B0** **H'FFB6** **8-Bit Timer Channel 0**
**TCORB1—Time Constant Register B1** **H'FFB7** **8-Bit Timer Channel 1**




**TCNT0—Timer Counter 0** **H'FFB8** **8-Bit Timer Channel 0**
**TCNT1—Timer Counter 1** **H'FFB9** **8-Bit Timer Channel 1**



| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OVF | WT/$\overline{IT}$ | TME | — | — | CKS2 | CKS1 | CKS0 |
| Initial value | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| Read/Write | R/(W)* | R/W | R/W | — | — | R/W | R/W | R/W |

Clock Select

| CKS2 | CKS1 | CKS0 | Clock | Overflow period* (when ø = 20 MHz) |
|---|---|---|---|---|
| 0 | 0 | 0 | ø/2 (initial value) | 25.6µs |
| | | 1 | ø/64 | 819.2µs |
| | 1 | 0 | ø/128 | 1.6ms |
| | | 1 | ø/512 | 6.6ms |
| 1 | 0 | 0 | ø/2048 | 26.2ms |
| | | 1 | ø/8192 | 104.9ms |
| | 1 | 0 | ø/32768 | 419.4ms |
| | | 1 | ø/131072 | 1.68s |

Note: * The overflow period is the time from when TCNT starts counting up from H'00 until overflow occurs.

Timer Enable

| | |
|---|---|
| 0 | TCNT is initialized to H'00 and halted |
| 1 | TCNT counts |

Timer Mode Select

| | |
|---|---|
| 0 | Interval timer mode: Sends the CPU an interval timer interrupt request (WOVI) when TCNT overflows |
| 1 | Watchdog timer mode: Generates the $\overline{WDTOVF}$ signal when TCNT overflows |

Overflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>Cleared by reading TCSR when OVF = 1, then writing 0 to OVF |
| 1 | [Setting condition]<br>Set when TCNT overflows from H'FF to H'00 in interval timer mode |

The method for writing to TCSR is different from that for general registers to prevent inadvertent overwriting. For details see section 11.2.4, Notes on Register Access.

Note: * Can only be written with 0 for flag clearing.

**TCNT—Timer Counter** **H'FFBC (W) H'FFBD (R)** **WDT**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The method for writing to TCNT is different from that for general registers to prevent inadvertent overwriting. For details, see section 11.2.4, Notes on Register Access.

**RSTCSR—Reset Control/Status Register** **H'FFBE (W) H'FFBF (R)** **WDT**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WOVF | RSTE | RSTS | — | — | — | — | — |
| Initial value | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | R/(W)* | R/W | R/W | — | — | — | — | — |

Reset Select

| | |
|---|---|
| 0 | Power-on reset |
| 1 | Manual reset |

Reset Enable

| | |
|---|---|
| 0 | Reset signal is not generated if TCNT overflows* |
| 1 | Reset signal is generated if TCNT overflows |

Note: * The modules H8S/2355 Series are not reset, but TCNT and TCSR in WDT are reset.

Watchdog Timer Overflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>Cleared by reading TCSR when WOVF = 1, then writing 0 to WOVF |
| 1 | [Setting condition]<br>Set when TCNT overflows (changed from H'FF to H'00) during watchdog timer operation |

Note: * Can only be written with 0 for flag clearing.

The method for writing to RSTCSR is different from that for general registers to prevent inadvertent overwriting. For details see section 11.2.4, Notes on Register Access.

**TSTR—Timer Start Register** **H'FFC0** **TPU**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | CST5 | CST4 | CST3 | CST2 | CST1 | CST0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

Counter Start

| | |
|---|---|
| 0 | TCNTn count operation is stopped |
| 1 | TCNTn performs count operation |

(n = 5 to 0)

Note: If 0 is written to the CST bit during operation with the TIOC pin designated for output, the counter stops but the TIOC pin output compare output level is retained. If TIOR is written to when the CST bit is cleared to 0, the pin output level will be changed to the set initial output value.

**TSYR—Timer Synchro Register** **H'FFC1** **TPU**

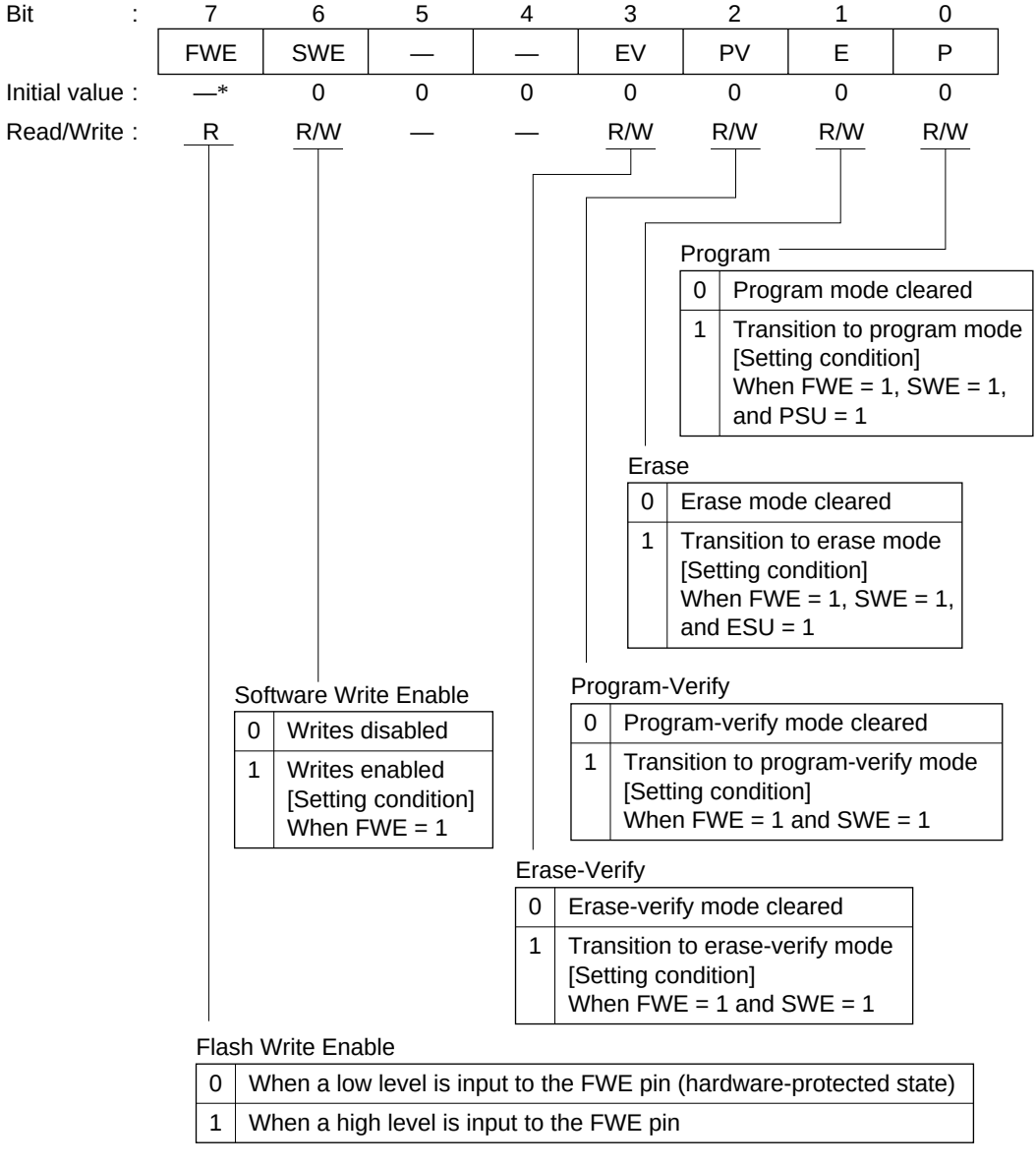
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | SYNC5 | SYNC4 | SYNC3 | SYNC2 | SYNC1 | SYNC0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

Timer Synchronization

| | |
|---|---|
| 0 | TCNTn operates independently (TCNT presetting/ clearing is unrelated to other channels) |
| 1 | TCNTn performs synchronous operation<br>TCNT synchronous presetting/synchronous clearing is possible |

(n = 5 to 0)

Notes: 1. To set synchronous operation, the SYNC bits for at least two channels must be set to 1.
2. To set synchronous clearing, in addition to the SYNC bit , the TCNT clearing source must also be set by means of bits CCLR2 to CCLR0 in TCR.




Note: * Determined by the state of the FWE pin.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FLER | — | — | — | — | — | ESU | PSU |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R | — | — | — | — | — | R/W | R/W |

Program Setup

| | |
|---|---|
| 0 | Program setup cleared |
| 1 | Program setup<br>[Setting condition]<br>When FWE = 1, and SWE = 1 |

Erase Setup

| | |
|---|---|
| 0 | Erase setup cleared |
| 1 | Erase setup<br>[Setting condition]<br>When FWE = 1, and SWE = 1 |

Flash Memory Error

| | |
|---|---|
| 0 | Flash memory is operating normally<br>Flash memory program/erase protection (error protection) is disabled<br>[Clearing condition]<br>Reset or hardware standby mode |
| 1 | An error has occurred during flash memory programming/erasing<br>Flash memory program/erase protection (error protection) is enabled<br>[Setting condition]<br>See section 19.10.3, Error Protection |

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| EBR1 | — | — | — | — | — | — | EB9 | EB8 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | — | — | — | — | — | — | R/W | R/W |

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| EBR2 | EB7 | EB6 | EB5 | EB4 | EB3 | EB2 | EB1 | EB0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Flash Memory Erase Blocks

| Block (Size) | Address |
|---|---|
| EB0 (1 kbyte) | H'000000 to H'0003FF |
| EB1 (1 kbyte) | H'000400 to H'0007FF |
| EB2 (1 kbyte) | H'000800 to H'000BFF |
| EB3 (1 kbyte) | H'000C00 to H'000FFF |
| EB4 (28 kbytes) | H'001000 to H'007FFF |
| EB5 (16 kbytes) | H'008000 to H'00BFFF |
| EB6 (8 kbytes) | H'00C000 to H'00DFFF |
| EB7 (8 kbytes) | H'00E000 to H'00FFFF |
| EB8 (32 kbytes) | H'010000 to H'017FFF |
| EB9 (32 kbytes) | H'018000 to H'01FFFF |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CCLR2 | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Time Prescaler

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | Internal clock: counts on ø/1 |
| | | 1 | Internal clock: counts on ø/4 |
| | 1 | 0 | Internal clock: counts on ø/16 |
| | | 1 | Internal clock: counts on ø/64 |
| 1 | 0 | 0 | External clock: counts on TCLKA pin input |
| | | 1 | External clock: counts on TCLKB pin input |
| | 1 | 0 | External clock: counts on TCLKC pin input |
| | | 1 | External clock: counts on TCLKD pin input |

Clock Edge

| | | |
|---|---|---|
| 0 | 0 | Count at rising edge |
| | 1 | Count at falling edge |
| 1 | * | Count at both edges |

*: Don't care

Counter Clear

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | TCNT clearing disabled |
| | | 1 | TCNT cleared by TGRA compare match/input capture |
| | 1 | 0 | TCNT cleared by TGRB compare match/input capture |
| | | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation*1 |
| 1 | 0 | 0 | TCNT clearing disabled |
| | | 1 | TCNT cleared by TGRC compare match/input capture*2 |
| | 1 | 0 | TCNT cleared by TGRD compare match/input capture*2 |
| | | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation*1 |

Notes: 1. Synchronous operation setting is performed by setting the SYNC bit in TSYR to 1.
2. When TGRC or TGRD is used as a buffer register, TCNT is not cleared because the buffer register setting has priority, and compare match/input capture does not occur.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | BFB | BFA | MD3 | MD2 | MD1 | MD0 |
| Initial value | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

Mode

| | | | | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Normal operation |
| | | | 1 | Reserved |
| | | 1 | 0 | PWM mode 1 |
| | | | 1 | PWM mode 2 |
| | 1 | 0 | 0 | Phase counting mode 1 |
| | | | 1 | Phase counting mode 2 |
| | | 1 | 0 | Phase counting mode 3 |
| | | | 1 | Phase counting mode 4 |
| 1 | * | * | * | — |

* : Don't care

Notes: 1. MD3 is a reserved bit. In a write, it should always be written with 0.
2. Phase counting mode cannot be set for channels 0 and 3. In this case, 0 should always be written to MD2.

TGRA Buffer Operation

| | |
|---|---|
| 0 | TGRA operates normally |
| 1 | TGRA and TGRC used together for buffer operation |

TGRB Buffer Operation

| | |
|---|---|
| 0 | TGRB operates normally |
| 1 | TGRB and TGRD used together for buffer operation |

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TGR0A I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR0A is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR0A is input capture register | Capture input source is TIOCA0 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is channel 1/count clock | Input capture at TCNT1 count-up/ count-down |

* : Don't care

TGR0B I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR0B is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR0B is input compare register | Capture input source is TIOCB0 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is channel 1/count clock | Input capture at TCNT1 count-up/ count-down*1 |

* : Don't care

Note: *1. When bits TPSC2 to TPSC0 in TCR1 are set to B'000, and ø/1 is used as the TCNT1 count clock, this setting is invalid and input capture is not generated.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | : | IOD3 | IOD2 | IOD1 | IOD0 | IOC3 | IOC2 | IOC1 | IOC0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TGR0C I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR0C is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR0C is input capture register *1 | Capture input source is TIOCC0 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is channel 1/count clock | Input capture at TCNT1 count-up/ count-down |

* : Don't care

Note: 1. When the BFA bit in TMDR0 is set to 1 and TGR0C is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

TGR0D I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR0D is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR0D is input capture register *2 | Capture input source is TIOCD0 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is channel 1/count clock | Input capture at TCNT1 count-up/ count-down*1 |

* : Don't care

Notes: 1. When bits TPSC2 to TPSC0 in TCR1 are set to B'000, and ø/1 is used as the TCNT1 count clock, this setting is invalid and input capture is not generated.
2. When the BFB bit in TMDR0 is set to 1 and TGR0D is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

Note: When TGRC or TGRD is designated for buffer operation, this setting is invalid and the register operates as a buffer register.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TTGE | — | — | TCIEV | TGIED | TGIEC | TGIEB | TGIEA |
| Initial value | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | — | — | R/W | R/W | R/W | R/W | R/W |

TGR Interrupt Enable A

| | |
|---|---|
| 0 | Interrupt requests (TGIA) by TGFA bit disabled |
| 1 | Interrupt requests (TGIA) by TGFA bit enabled |

TGR Interrupt Enable B

| | |
|---|---|
| 0 | Interrupt requests (TGIB) by TGFB bit disabled |
| 1 | Interrupt requests (TGIB) by TGFB bit enabled |

TGR Interrupt Enable C

| | |
|---|---|
| 0 | Interrupt requests (TGIC) by TGFC bit disabled |
| 1 | Interrupt requests (TGIC) by TGFC bit enabled |

TGR Interrupt Enable D

| | |
|---|---|
| 0 | Interrupt requests (TGID) by TGFD bit disabled |
| 1 | Interrupt requests (TGID) by TGFD bit enabled |

Overflow Interrupt Enable

| | |
|---|---|
| 0 | Interrupt requests (TCIV) by TCFV disabled |
| 1 | Interrupt requests (TCIV) by TCFV enabled |

A/D Conversion Start Request Enable

| | |
|---|---|
| 0 | A/D conversion start request generation disabled |
| 1 | A/D conversion start request generation enabled |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | TCFV | TGFD | TGFC | TGFB | TGFA |
| Initial value | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | — | — | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* |

Input Capture/Output Compare Flag A

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIA interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFA after reading TGFA = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRA while TGRA is functioning as output compare register<br>• When TCNT value is transferred to TGRA by input capture signal while TGRA is functioning as input capture register |

Input Capture/Output Compare Flag B

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIB interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFB after reading TGFB = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRB while TGRB is functioning as output compare register<br>• When TCNT value is transferred to TGRB by input capture signal while TGRB is functioning as input capture register |

Input Capture/Output Compare Flag C

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIC interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFC after reading TGFC = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRC while TGRC is functioning as output compare register<br>• When TCNT value is transferred to TGRC by input capture signal while TGRC is functioning as input capture register |

Input Capture/Output Compare Flag D

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGID interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFD after reading TGFD = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRD while TGRD is functioning as output compare register<br>• When TCNT value is transferred to TGRD by input capture signal while TGRD is functioning as input capture register |

Overflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFV after reading TCFV = 1 |
| 1 | [Setting conditions]<br>When the TCNT value overflows (changes from H'FFFF to H'0000 ) |

Note: * Can only be written with 0 for flag clearing.

**TCNT0—Timer Counter 0** **H'FFD6** **TPU0**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Up-counter

**TGR0A—Timer General Register 0A** **H'FFD8** **TPU0**
**TGR0B—Timer General Register 0B** **H'FFDA** **TPU0**
**TGR0C—Timer General Register 0C** **H'FFDC** **TPU0**
**TGR0D—Timer General Register 0D** **H'FFDE** **TPU0**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write : | — | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Time Prescaler

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | Internal clock: counts on ø/1 |
| | | 1 | Internal clock: counts on ø/4 |
| | 1 | 0 | Internal clock: counts on ø/16 |
| | | 1 | Internal clock: counts on ø/64 |
| 1 | 0 | 0 | External clock: counts on TCLKA pin input |
| | | 1 | External clock: counts on TCLKB pin input |
| | 1 | 0 | Internal clock: counts on ø/256 |
| | | 1 | Counts on TCNT2 overflow/underflow |

Note: This setting is ignored when channel 1 is in phase counting mode.

Clock Edge

| | | |
|---|---|---|
| 0 | 0 | Count at rising edge |
| | 1 | Count at falling edge |
| 1 | * | Count at both edges |

*: Don't care

Note: This setting is ignored when channel 1 is in phase counting mode.

Counter Clear

| | | |
|---|---|---|
| 0 | 0 | TCNT clearing disabled |
| | 1 | TCNT cleared by TGRA compare match/input capture |
| 1 | 0 | TCNT cleared by TGRB compare match/input capture |
| | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation* |

Note: * Synchronous operating setting is performed by setting the SYNC bit in TSYR to 1.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | MD3 | MD2 | MD1 | MD0 |
| Initial value | : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | — | — | — | — | R/W | R/W | R/W | R/W |

Mode

| | | | | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Normal operation |
| | | | 1 | Reserved |
| | | 1 | 0 | PWM mode 1 |
| | | | 1 | PWM mode 2 |
| | 1 | 0 | 0 | Phase counting mode 1 |
| | | | 1 | Phase counting mode 2 |
| | | 1 | 0 | Phase counting mode 3 |
| | | | 1 | 1Phase counting mode 4 |
| 1 | * | * | * | — |

* : Don't care

Notes: MD3 is a reserved bit. In a write, it should always be written with 0.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TGR1A I/O Control

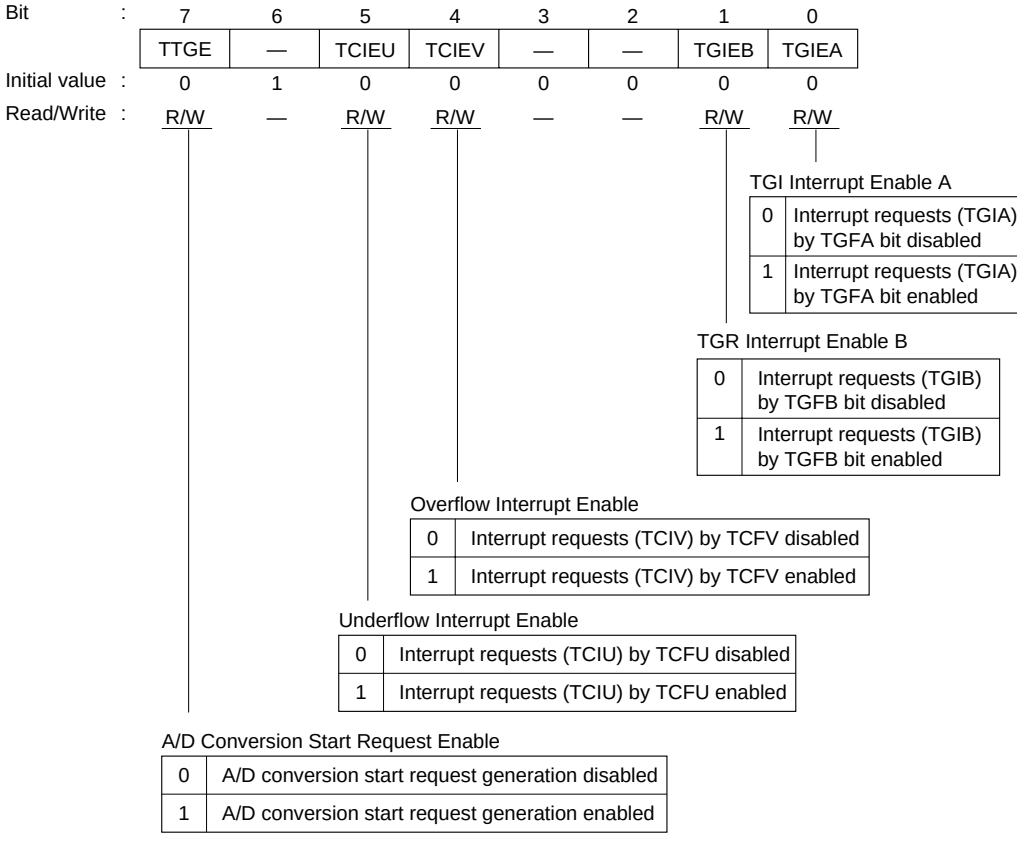
| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR1A is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR1A is input capture register | Capture input source is TIOCA1 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is TGR0A compare match/ input capture | Input capture at generation of channel 0/TGR0A compare match/ input capture |

* : Don't care

TGR1B I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR1B is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | 0 | 0 | 0 | TGR1B is input capture register | Capture input source is TIOCB1 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |
| | 1 | * | * | | Capture input source is TGR0C compare match/ input capture | Input capture at generation of TGR0B compare match/input capture |

* : Don't care


Bit : 7 6 5 4 3 2 1 0
TTGE — TCIEU TCIEV — — TGIEB TGIEA
Initial value : 0 1 0 0 0 0 0 0
Read/Write : R/W — R/W R/W — — R/W R/W
TGI Interrupt Enable A
0 Interrupt requests (TGIA) by TGFA bit disabled
1 Interrupt requests (TGIA) by TGFA bit enabled
TGR Interrupt Enable B
0 Interrupt requests (TGIB) by TGFB bit disabled
1 Interrupt requests (TGIB) by TGFB bit enabled
Overflow Interrupt Enable
0 Interrupt requests (TCIV) by TCFV disabled
1 Interrupt requests (TCIV) by TCFV enabled
Underflow Interrupt Enable
0 Interrupt requests (TCIU) by TCFU disabled
1 Interrupt requests (TCIU) by TCFU enabled
A/D Conversion Start Request Enable
0 A/D conversion start request generation disabled
1 A/D conversion start request generation enabled

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA |
| Initial value | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R | — | R/(W)* | R/(W)* | — | — | R/(W)* | R/(W)* |

Input Capture/Output Compare Flag A

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIA interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFA after reading TGFA = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRA while TGRA is functioning as output compare register<br>• When TCNT value is transferred to TGRA by input capture signal while TGRA is functioning as input capture register |

Input Capture/Output Compare Flag B

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIB interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFB after reading TGFB = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRB while TGRB is functioning as output compare register<br>• When TCNT value is transferred to TGRB by input capture signal while TGRB is functioning as input capture register |

Overflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFV after reading TCFV = 1 |
| 1 | [Setting conditions]<br>When the TCNT value overflows (changes from H'FFFF to H'0000 ) |

Underflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFU after reading TCFU = 1 |
| 1 | [Setting conditions]<br>When the TCNT value underflows (changes from H'0000 to H'FFFF) |

Count Direction Flag

| | |
|---|---|
| 0 | TCNT counts down |
| 1 | TCNT counts up |

Note: * Can only be written with 0 for flag clearing.

**TCNT1—Timer Counter 1** **H'FFE6** **TPU1**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Up/down-counter*

Note: * This timer counter can be used as an up/down-counter only in phase counting mode or when performing overflow/underflow counting on another channel. In other cases it functions as an up-counter.

**TGR1A—Timer General Register 1A** **H'FFE8** **TPU1**
**TGR1B—Timer General Register 1B** **H'FFEA** **TPU1**

| Bit | : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | |
| Initial value | : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | — | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Time Prescaler

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | Internal clock: counts on ø/1 |
| | | 1 | Internal clock: counts on ø/4 |
| | 1 | 0 | Internal clock: counts on ø/16 |
| | | 1 | Internal clock: counts on ø/64 |
| 1 | 0 | 0 | External clock: counts on TCLKA pin input |
| | | 1 | External clock: counts on TCLKB pin input |
| | 1 | 0 | External clock: counts on TCLKC pin input |
| | | 1 | Internal clock: counts on ø/1024 |

Note: This setting is ignored when channel 2 is in phase counting mode.

Clock Edge

| | | |
|---|---|---|
| 0 | 0 | Count at rising edge |
| | 1 | Count at falling edge |
| 1 | * | Count at both edges |

*: Don't care

Note: This setting is ignored when channel 2 is in phase counting mode.

Counter Clear

| | | |
|---|---|---|
| 0 | 0 | TCNT clearing disabled |
| | 1 | TCNT cleared by TGRA compare match/input capture |
| 1 | 0 | TCNT cleared by TGRB compare match/input capture |
| | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation* |

Note: * Synchronous operating setting is performed by setting the SYNC bit TSYR to 1.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | — | — | — | — | MD3 | MD2 | MD1 | MD0 |
| Initial value | : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | — | — | — | — | R/W | R/W | R/W | R/W |

Mode

| | | | | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Normal operation |
| | | | 1 | Reserved |
| | | 1 | 0 | PWM mode 1 |
| | | | 1 | PWM mode 2 |
| | 1 | 0 | 0 | Phase counting mode 1 |
| | | | 1 | Phase counting mode 2 |
| | | 1 | 0 | Phase counting mode 3 |
| | | | 1 | Phase counting mode 4 |
| 1 | * | * | * | — |

* : Don't care

Notes: MD3 is a reserved bit. In a write, it should always be written with 0.

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TGR2A I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR2A is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | * | 0 | 0 | TGR2A is input capture register | Capture input source is TIOCA2 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |

* : Don't care

TGR2B I/O Control

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | TGR2B is output compare register | Output disabled | |
| | | | 1 | | Initial output is 0 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled | |
| | | | 1 | | Initial output is 1 output | 0 output at compare match |
| | | 1 | 0 | | | 1 output at compare match |
| | | | 1 | | | Toggle output at compare match |
| 1 | * | 0 | 0 | TGR2B is input capture register | Capture input source is TIOCB2 pin | Input capture at rising edge |
| | | | 1 | | | Input capture at falling edge |
| | | 1 | * | | | Input capture at both edges |

* : Don't care

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | TTGE | — | TCIEU | TCIEV | — | — | TGIEB | TGIEA |
| Initial value | : | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R/W | — | R/W | R/W | — | — | R/W | R/W |

TGR Interrupt Enable A

| | |
|---|---|
| 0 | Interrupt requests (TGIA) by TGFA bit disabled |
| 1 | Interrupt requests (TGIA) by TGFA bit enabled |

TGR Interrupt Enable B

| | |
|---|---|
| 0 | Interrupt requests (TGIB) by TGFB bit disabled |
| 1 | Interrupt requests (TGIB) by TGFB bit enabled |

Overflow Interrupt Enable

| | |
|---|---|
| 0 | Interrupt requests (TCIV) by TCFV disabled |
| 1 | Interrupt requests (TCIV) by TCFV enabled |

Underflow Interrupt Enable

| | |
|---|---|
| 0 | Interrupt requests (TCIU) by TCFU disabled |
| 1 | Interrupt requests (TCIU) by TCFU enabled |

A/D Conversion Start Request Enable

| | |
|---|---|
| 0 | A/D conversion start request generation disabled |
| 1 | A/D conversion start request generation enabled |

| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA |
| Initial value | : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | : | R | — | R/(W)* | R/(W)* | — | — | R/(W)* | R/(W)* |

Input Capture/Output Compare Flag A

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIA interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFA after reading TGFA = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRA while TGRA is functioning as output compare register<br>• When TCNT value is transferred to TGRA by input capture signal while TGRA is functioning as input capture register |

Input Capture/Output Compare Flag B

| | |
|---|---|
| 0 | [Clearing condition]<br>• When DTC is activated by TGIB interrupt while DISEL bit of MRB in DTC is 0<br>• When 0 is written to TGFB after reading TGFB = 1 |
| 1 | [Setting conditions]<br>• When TCNT = TGRB while TGRB is functioning as output compare register<br>• When TCNT value is transferred to TGRB by input capture signal while TGRB is functioning as input capture register |

Overflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFV after reading TCFV = 1 |
| 1 | [Setting conditions]<br>When the TCNT value overflows (changes from H'FFFF to H'0000 ) |

Underflow Flag

| | |
|---|---|
| 0 | [Clearing condition]<br>When 0 is written to TCFU after reading TCFU = 1 |
| 1 | [Setting conditions]<br>When the TCNT value underflows (changes from H'0000 to H'FFFF) |

Count Direction Flag

| | |
|---|---|
| 0 | TCNT counts down |
| 1 | TCNT counts up |

Note: * Can only be written with 0 for flag clearing.

**TCNT2—Timer Counter 2** **H'FFF6** **TPU2**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Up/down-counter*

Note: * This timer counter can be used as an up/down-counter only in phase counting mode or when performing overflow/underflow counting on another channel. In other cases it functions as an up-counter.

**TGR2A—Timer General Register 2A** **H'FFF8** **TPU2**
**TGR2B—Timer General Register 2B** **H'FFFA** **TPU2**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Initial value | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

## C.1 Port 1 Block Diagram

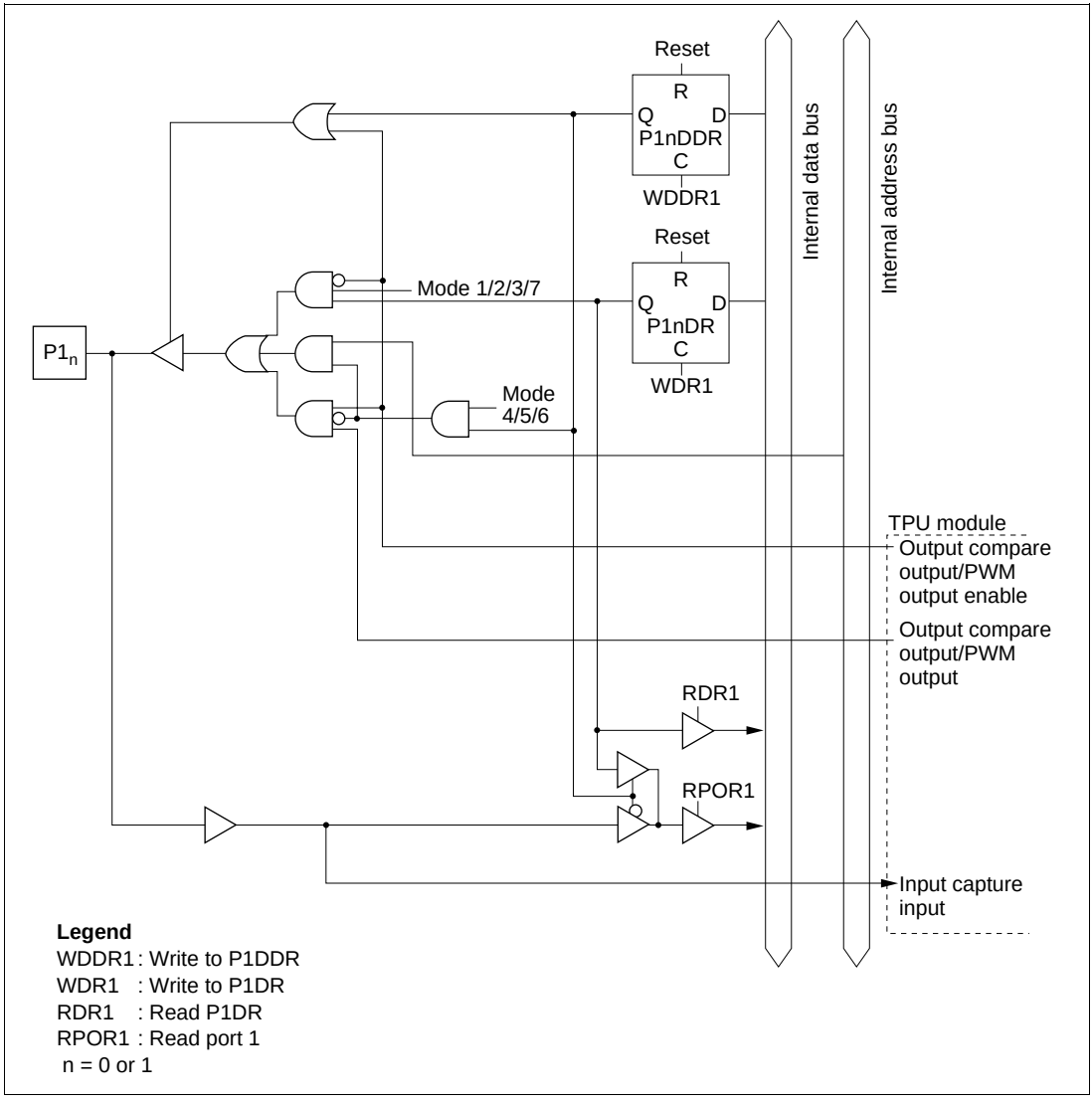



**Figure C.1 (a) Port 1 Block Diagram (Pins $P1_0$ and $P1_1$)**

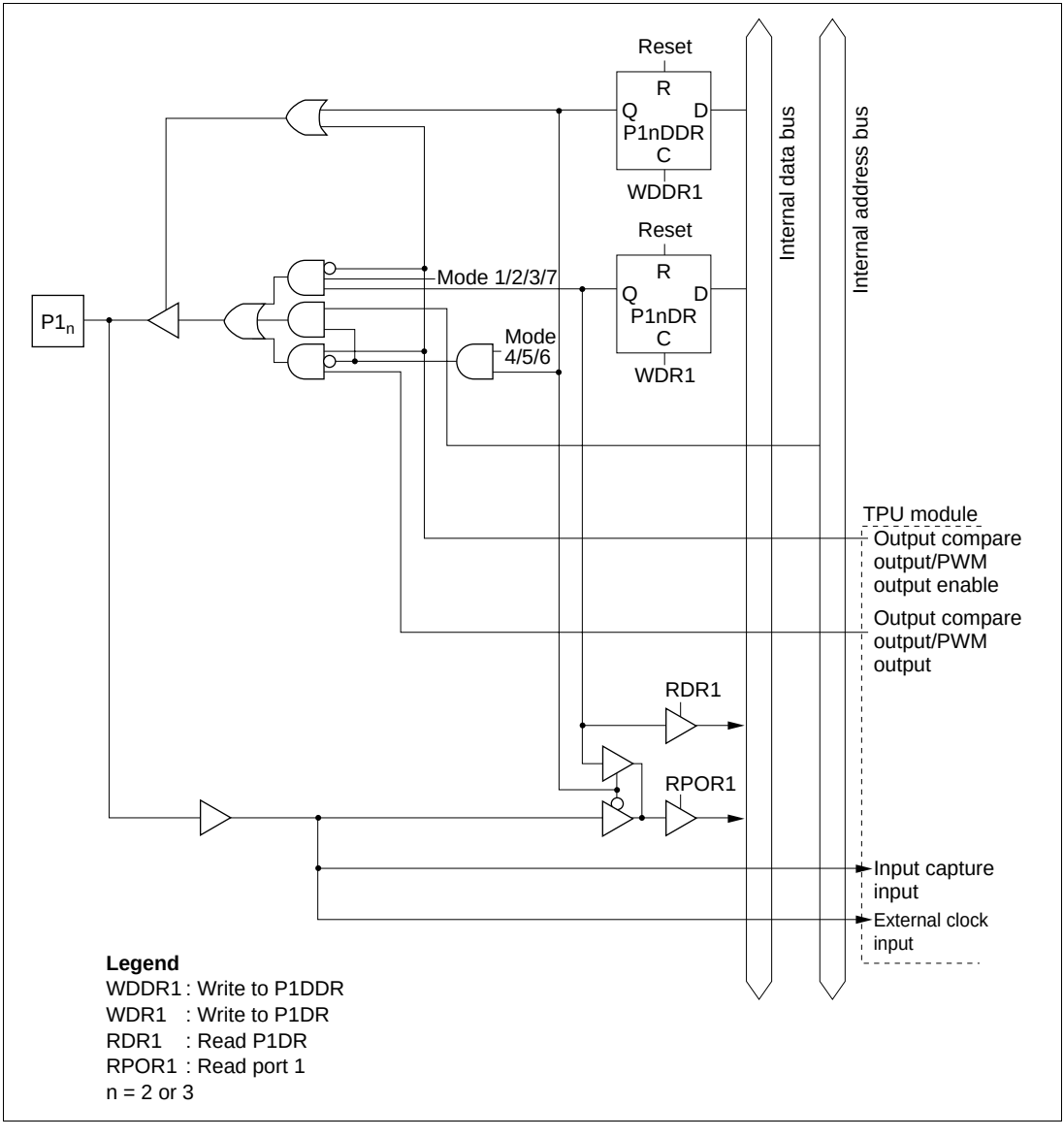



**Figure C.1 (b) Port 1 Block Diagram (Pins $P1_2$ and $P1_3$)**

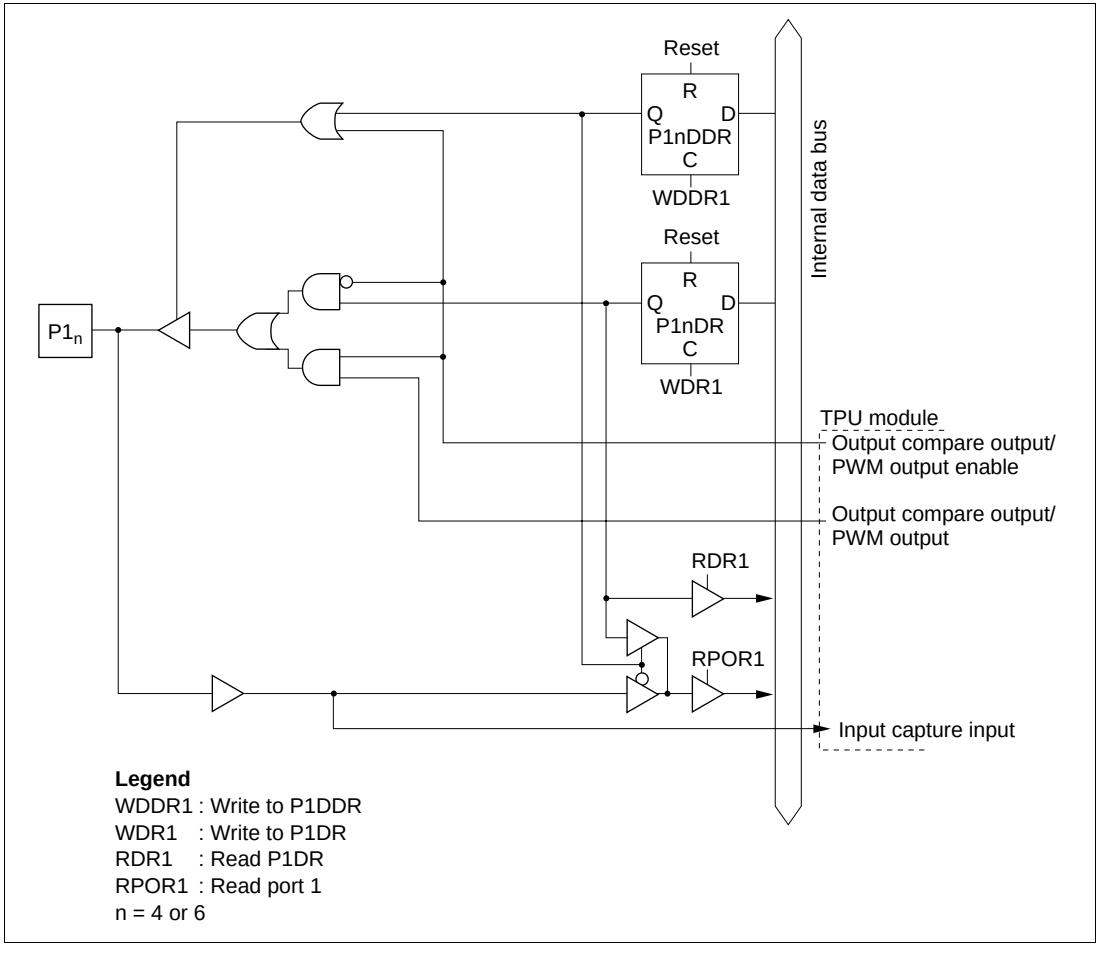



**Figure C.1 (c) Port 1 Block Diagram (Pins $P1_4$ and $P1_6$)**

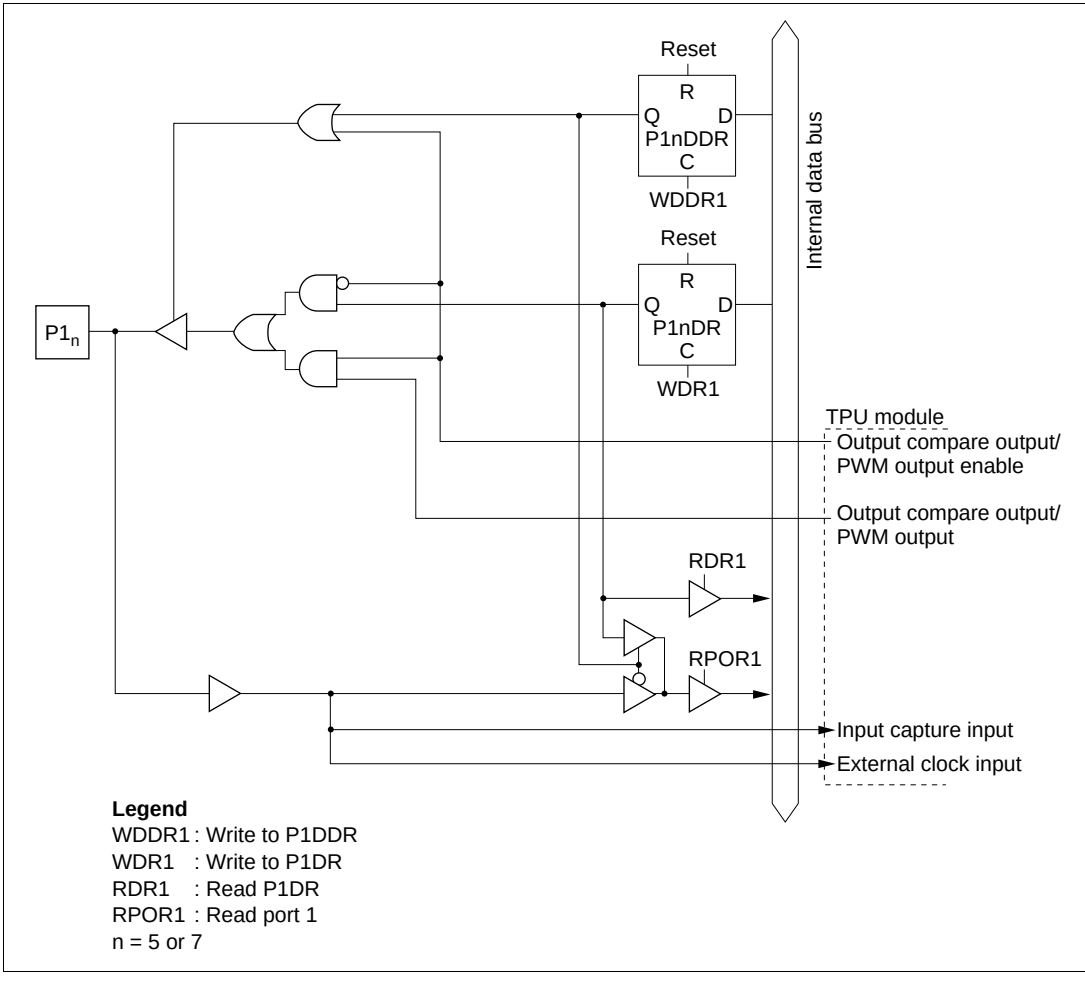



**Figure C.1 (d) Port 1 Block Diagram (Pins $P1_5$ and $P1_7$)**

## C.2 Port 2 Block Diagram

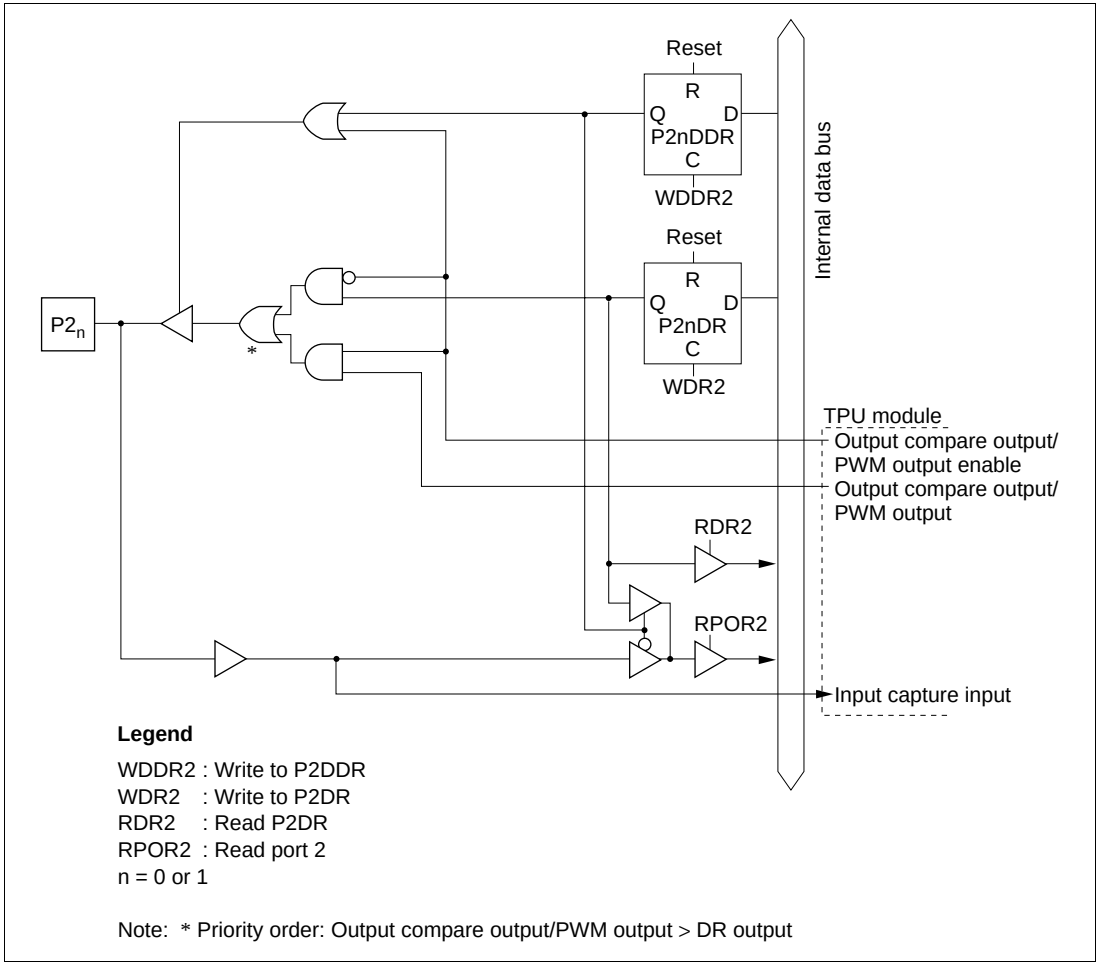
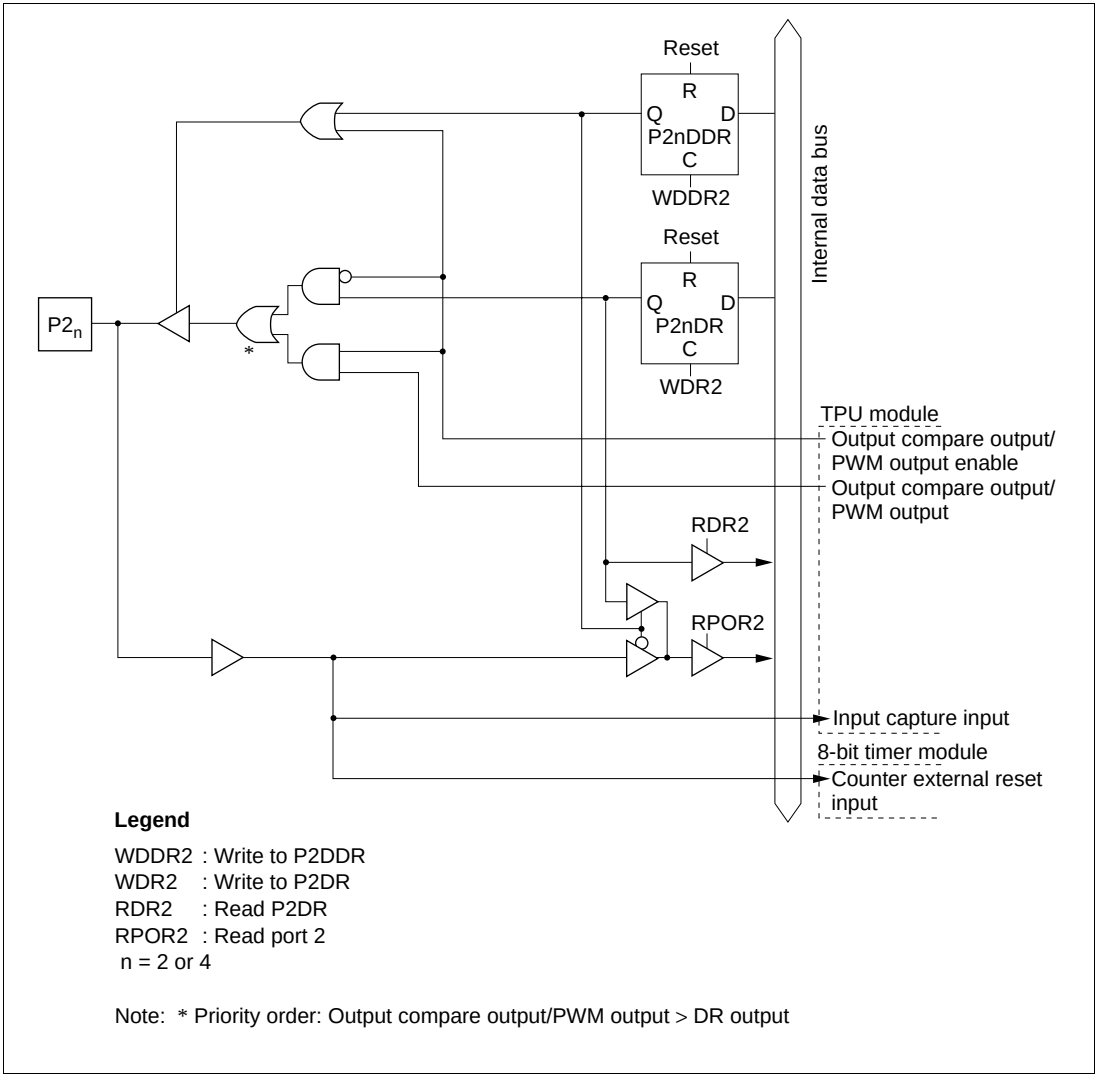



**Figure C.2 (a) Port 2 Block Diagram (Pins $P2_0$ and $P2_1$)**




**Legend**

WDDR2 : Write to P2DDR
WDR2 : Write to P2DR
RDR2 : Read P2DR
RPOR2 : Read port 2
n = 2 or 4

Note: * Priority order: Output compare output/PWM output > DR output

**Figure C.2 (b) Port 2 Block Diagram (Pins $P2_2$ and $P2_4$)**

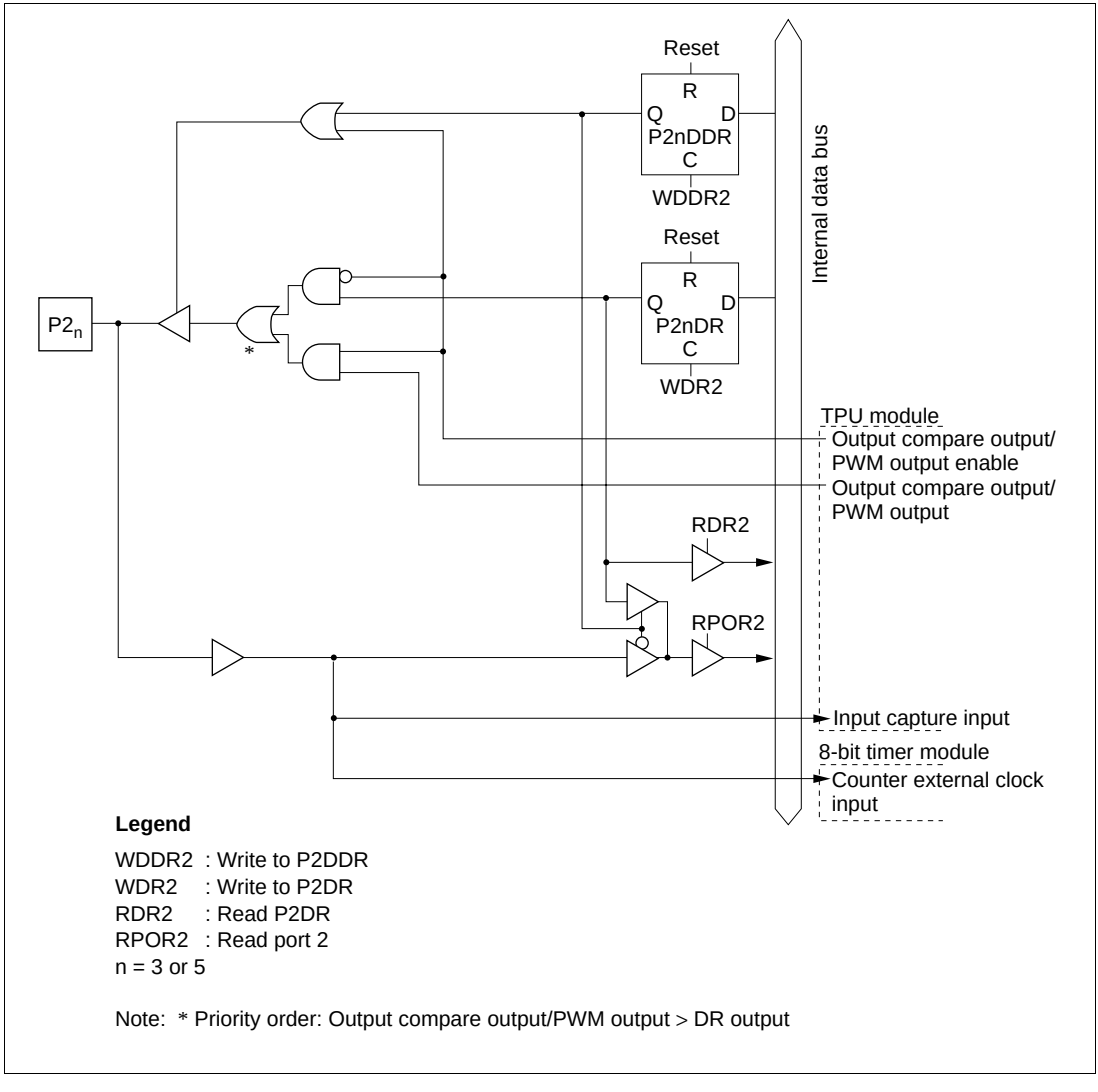
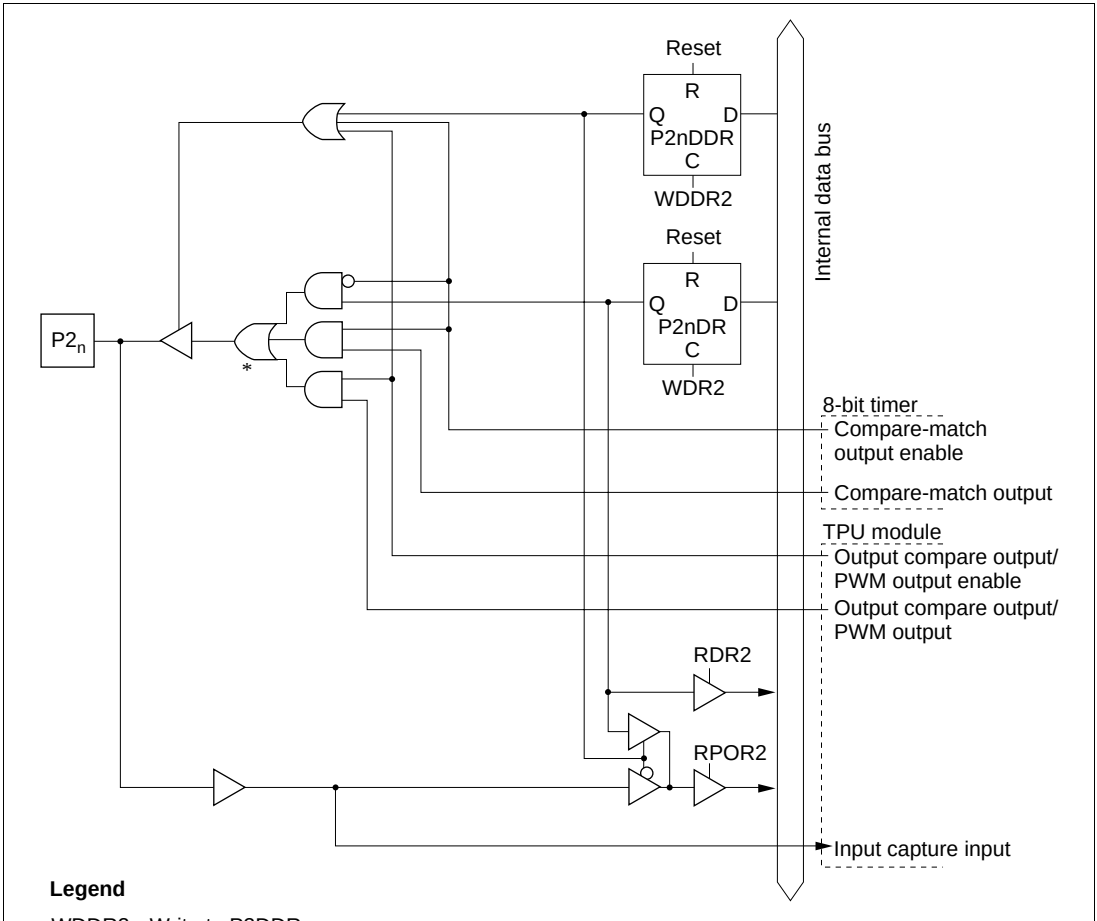



**Figure C.2 (c) Port 2 Block Diagram (Pins $P2_3$ and $P2_5$)**



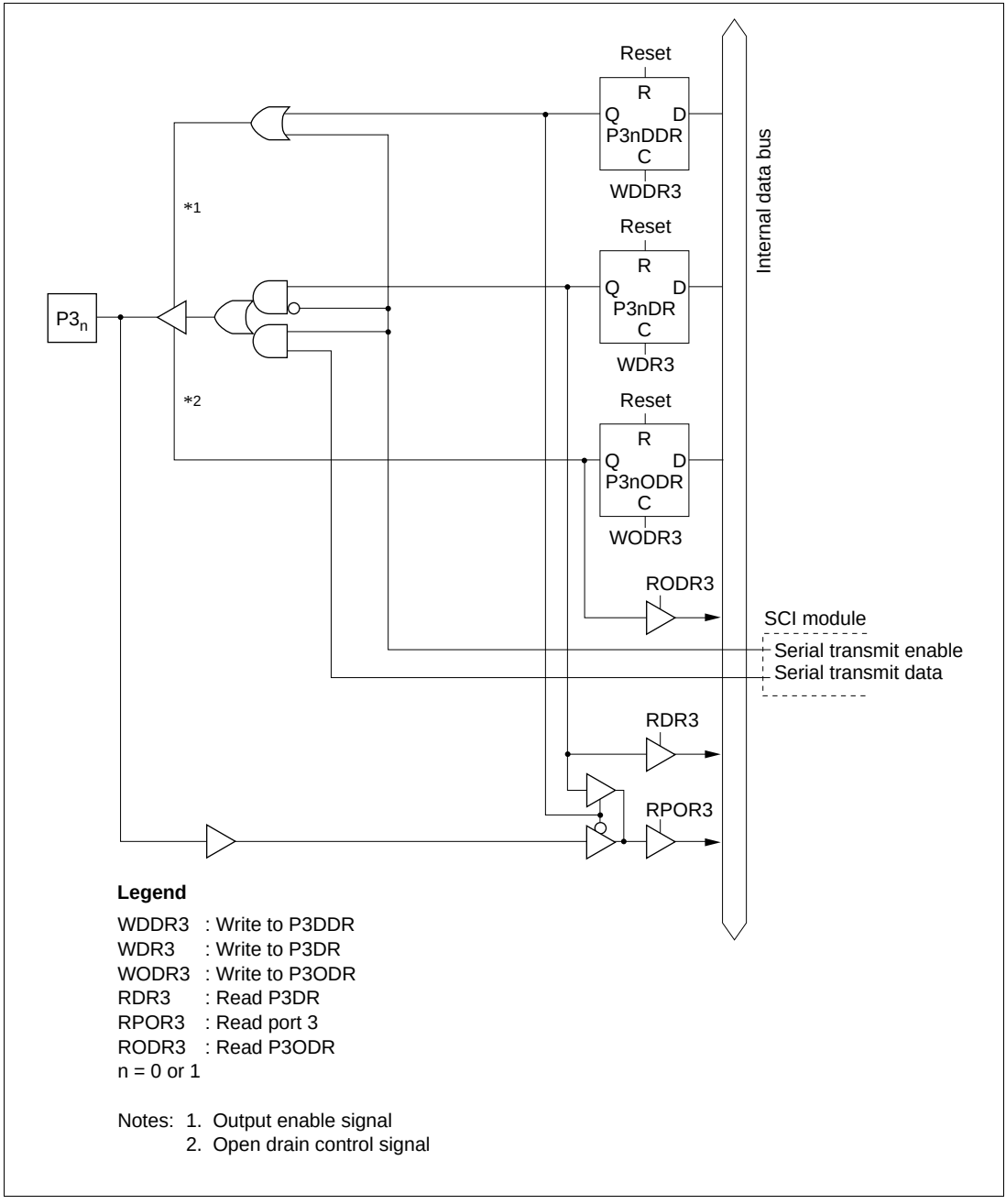

**Legend**

WDDR2 : Write to P2DDR
WDR2 : Write to P2DR
RDR2 : Read P2DR
RPOR2 : Read port 2
n = 6 or 7

Note: * Priority order:
Output compare output/PWM output > compare match output > DR output

**Figure C.2 (d) Port 2 Block Diagram (Pins $P2_6$ and $P2_7$)**

## C.3 Port 3 Block Diagram

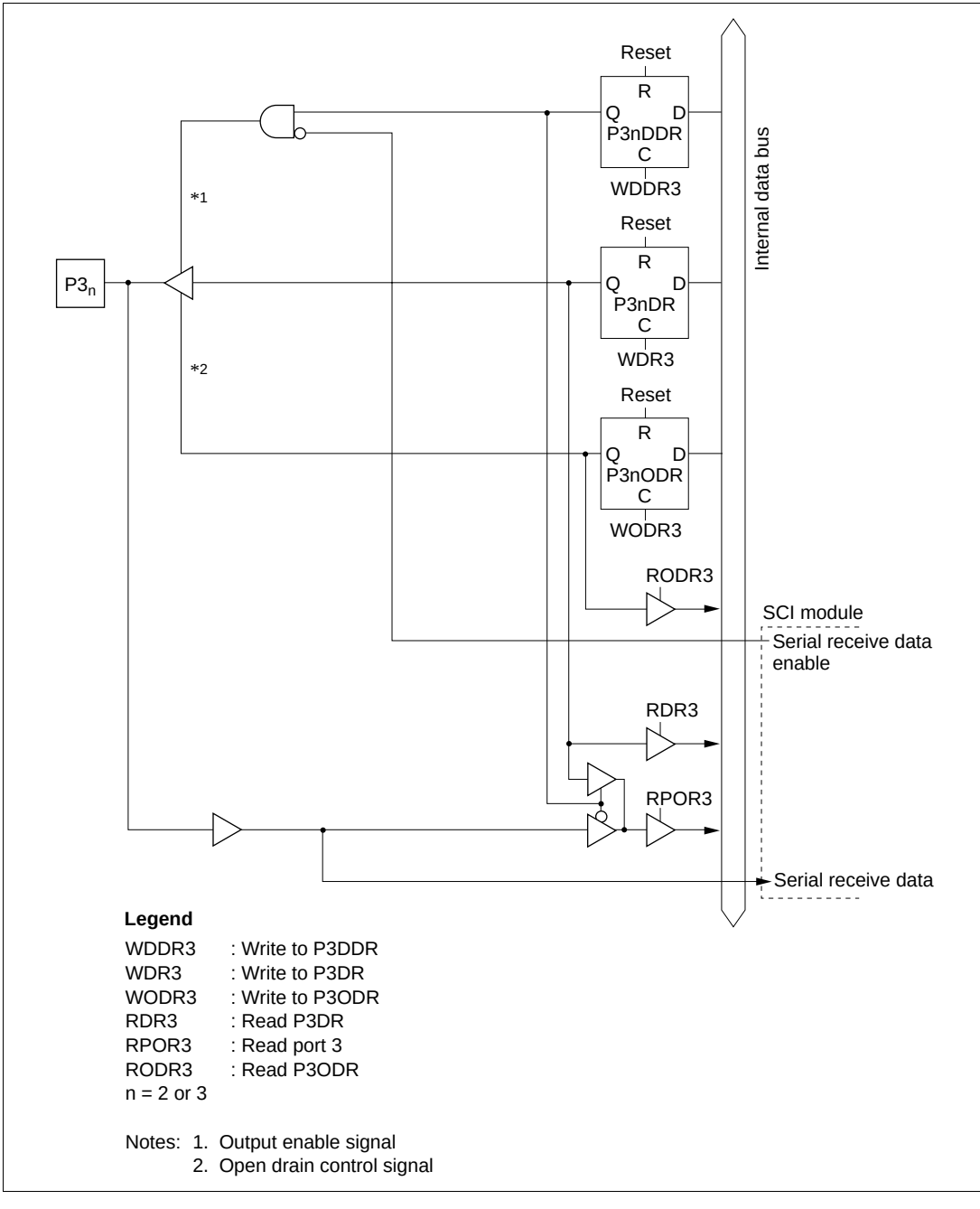



**Legend**

WDDR3 : Write to P3DDR
WDR3 : Write to P3DR
WODR3 : Write to P3ODR
RDR3 : Read P3DR
RPOR3 : Read port 3
RODR3 : Read P3ODR
n = 0 or 1

Notes: 1. Output enable signal
2. Open drain control signal

**Figure C.3 (a) Port 3 Block Diagram (Pins $P3_0$ and $P3_1$)**

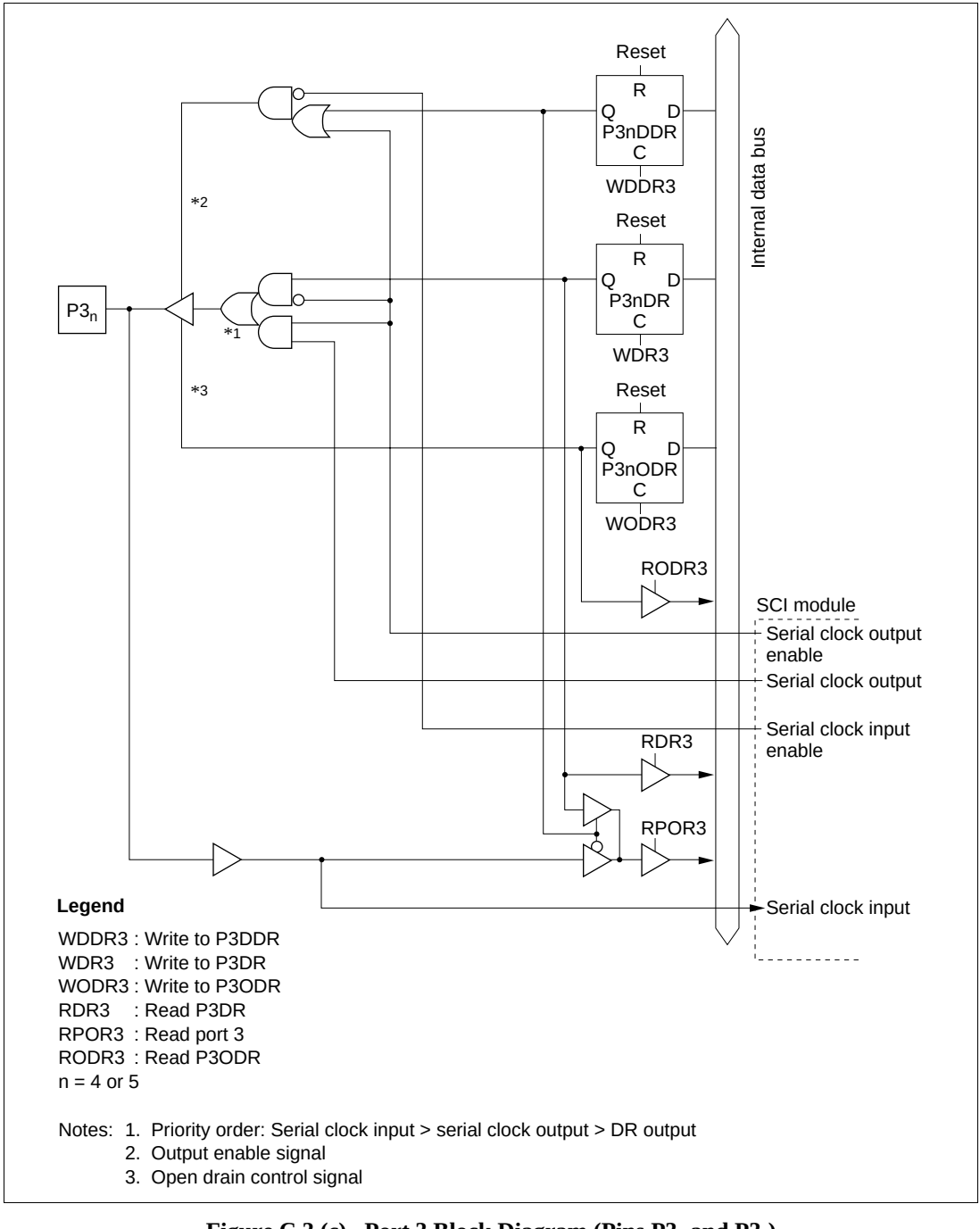



**Legend**

WDDR3 : Write to P3DDR
WDR3 : Write to P3DR
WODR3 : Write to P3ODR
RDR3 : Read P3DR
RPOR3 : Read port 3
RODR3 : Read P3ODR
n = 2 or 3

Notes: 1. Output enable signal
2. Open drain control signal

**Figure C.3 (b) Port 3 Block Diagram (Pins $P3_2$ and $P3_3$)**




**Legend**

WDDR3 : Write to P3DDR
WDR3 : Write to P3DR
WODR3 : Write to P3ODR
RDR3 : Read P3DR
RPOR3 : Read port 3
RODR3 : Read P3ODR
n = 4 or 5

Notes: 1. Priority order: Serial clock input > serial clock output > DR output
2. Output enable signal
3. Open drain control signal

**Figure C.3 (c) Port 3 Block Diagram (Pins $P3_4$ and $P3_5$)**

## C.4 Port 4 Block Diagram

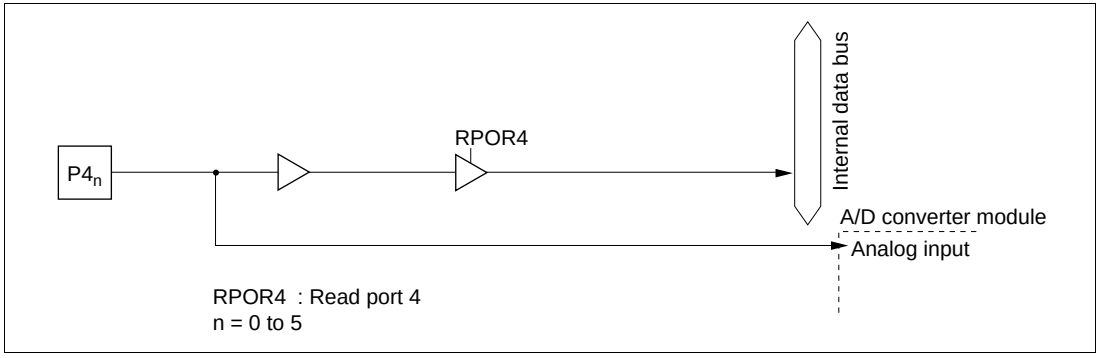



**Figure C.4 (a) Port 4 Block Diagram (Pins $P4_0$ to $P4_5$)**

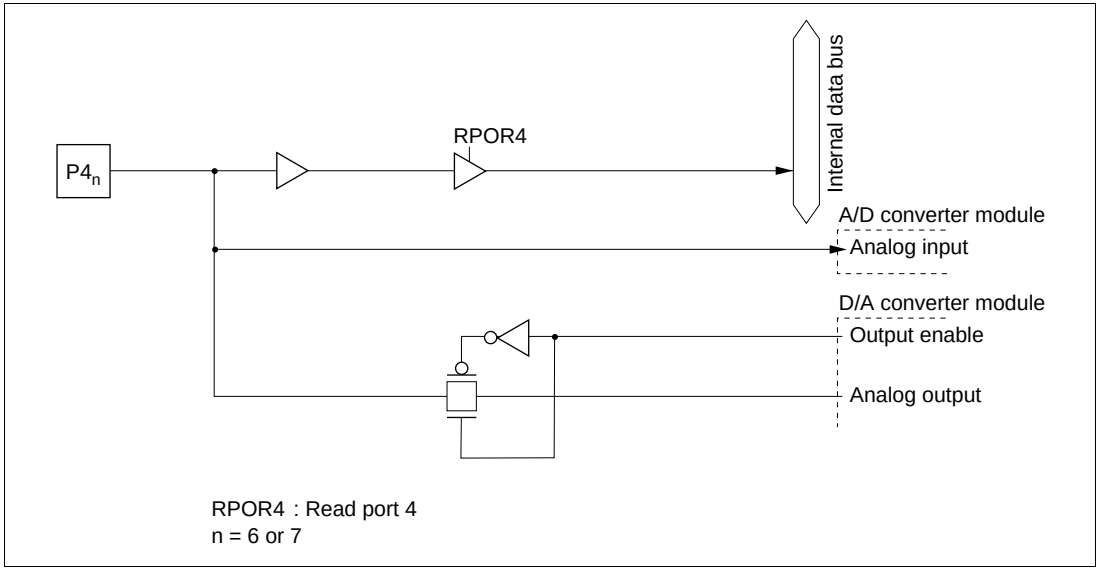



**Figure C.4 (b) Port 4 Block Diagram (Pins $P4_6$ and $P4_7$)**

## C.5 Port A Block Diagram

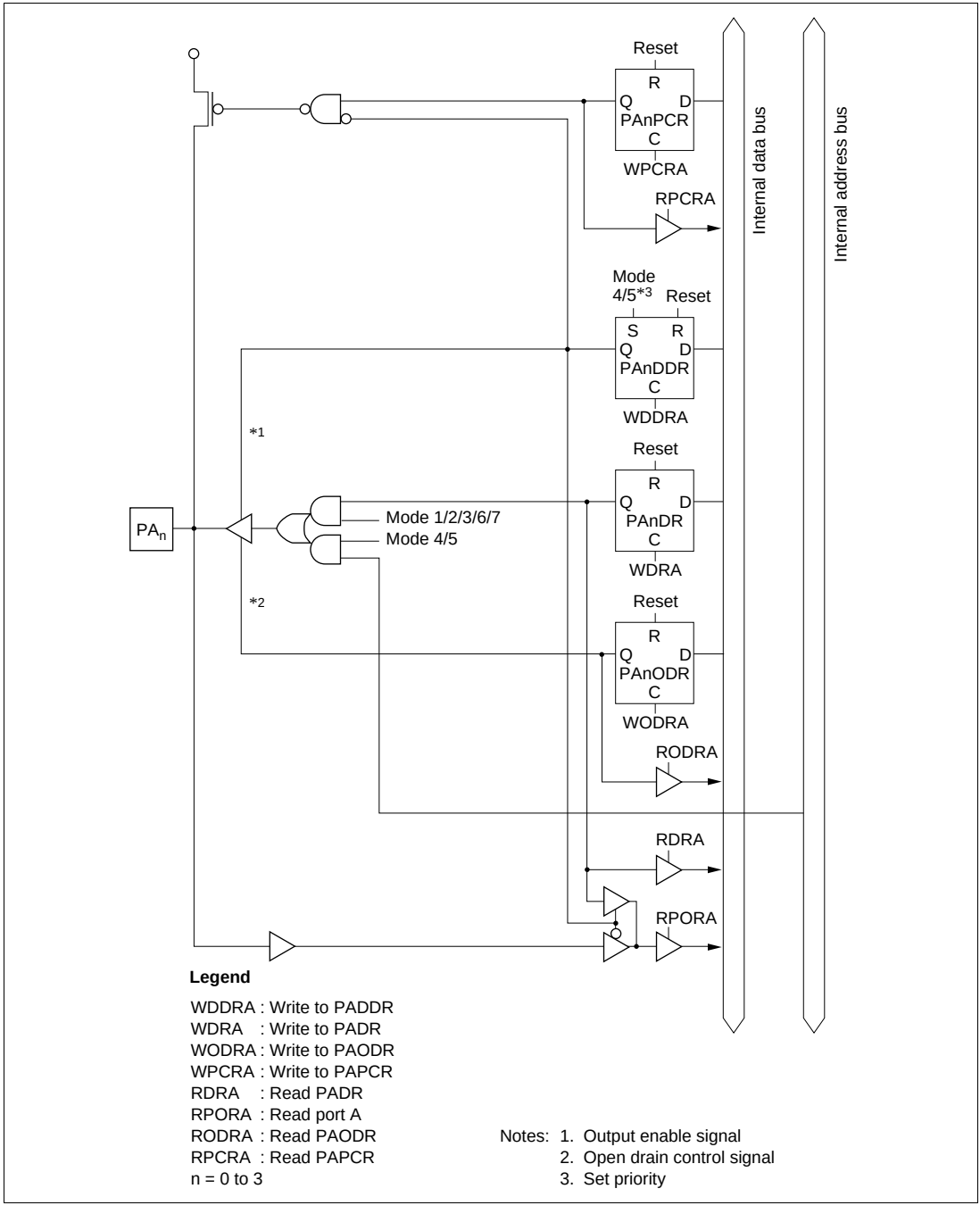



**Figure C.5 Port A Block Diagram (Pins $PA_0$ to $PA_3$)**

## C.6 Port B Block Diagram

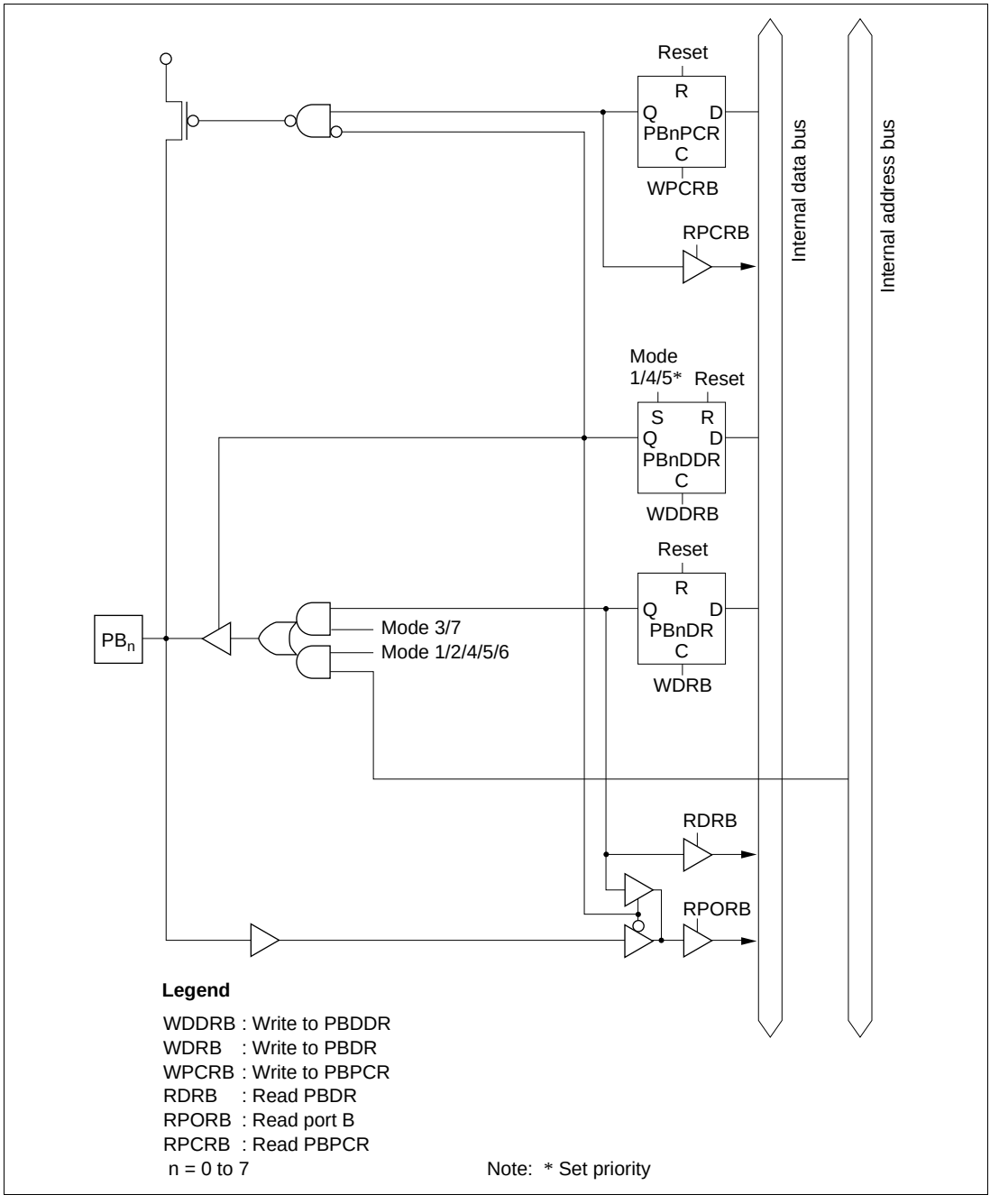



**Figure C.6 Port B Block Diagram (Pin $PB_n$)**

## C.7 Port C Block Diagram

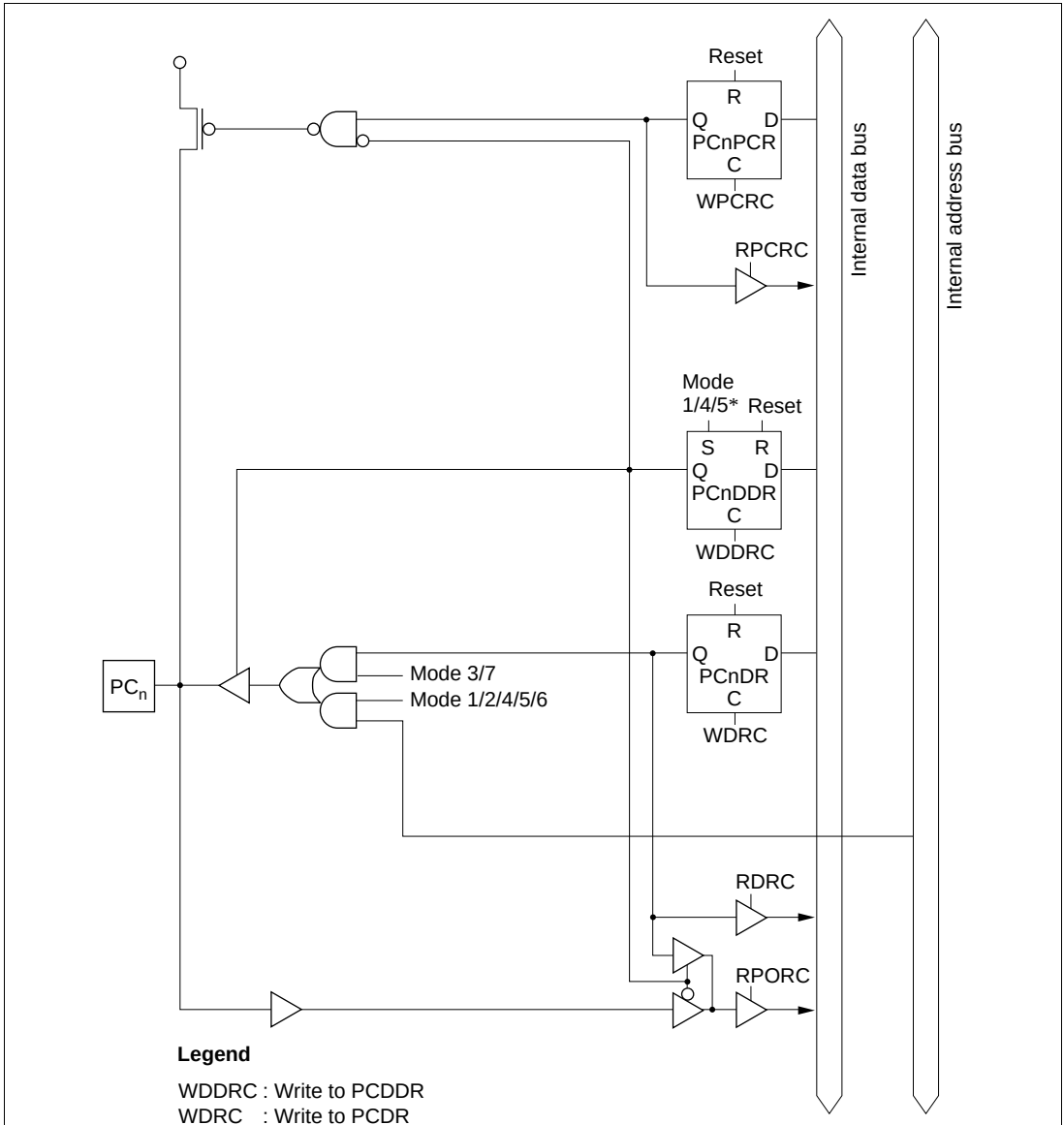



**Figure C.7 Port C Block Diagram (Pin $PC_n$)**

## C.8 Port D Block Diagram

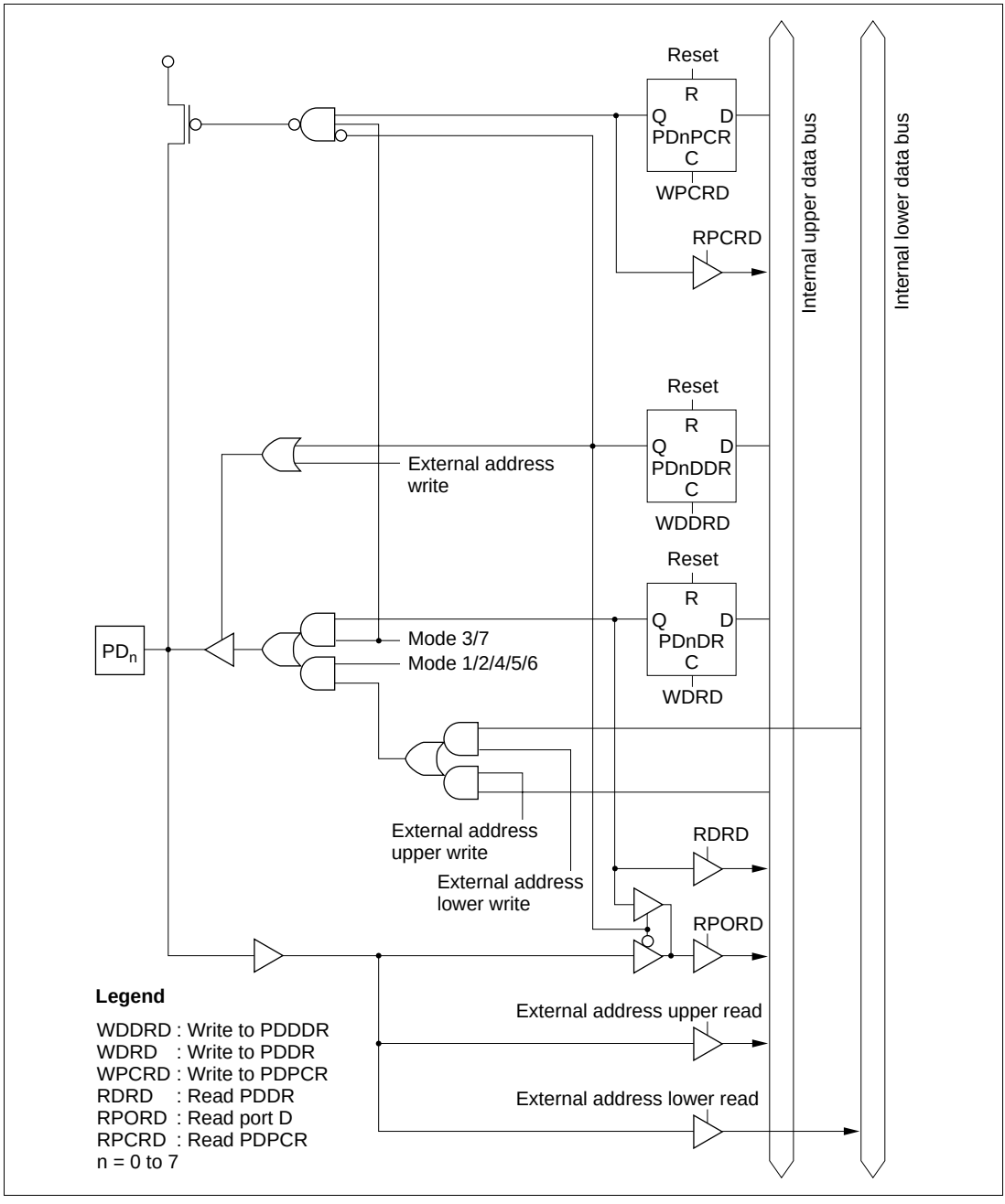



**Figure C.8 Port D Block Diagram (Pin $PD_n$)**

## C.9 Port E Block Diagram

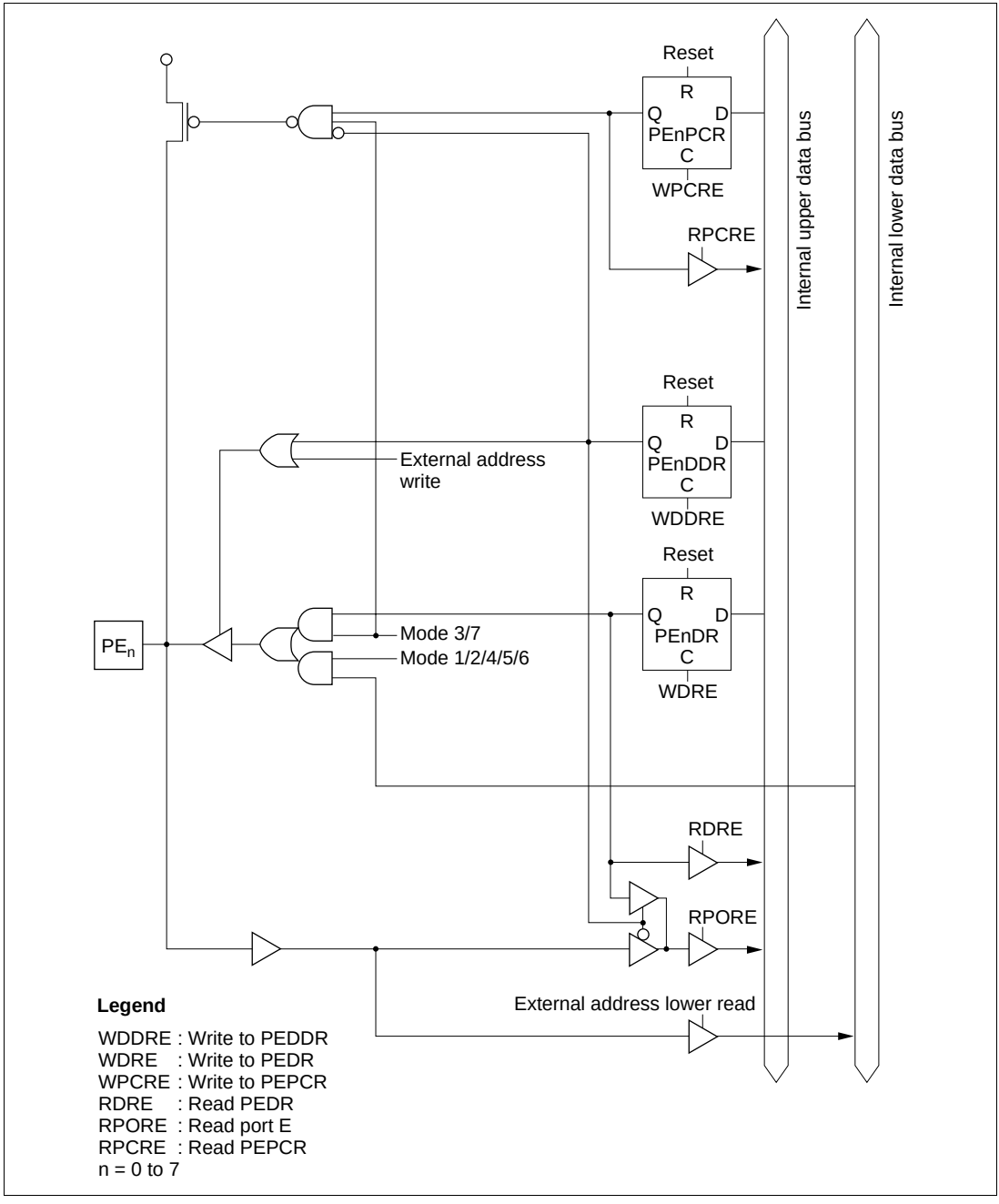



**Figure C.9 Port E Block Diagram (Pin $PE_n$)**

## C.10 Port F Block Diagram

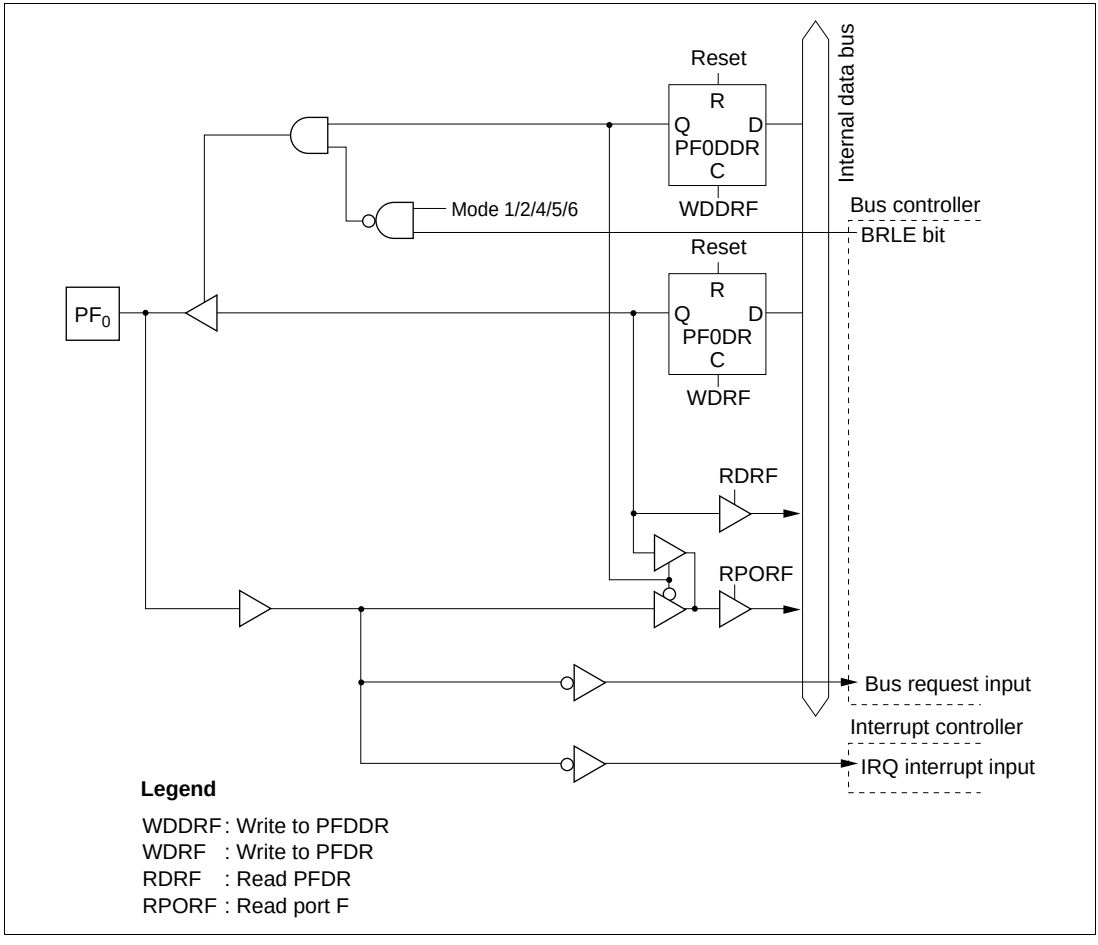



**Figure C.10 (a) Port F Block Diagram (Pin $PF_0$)**

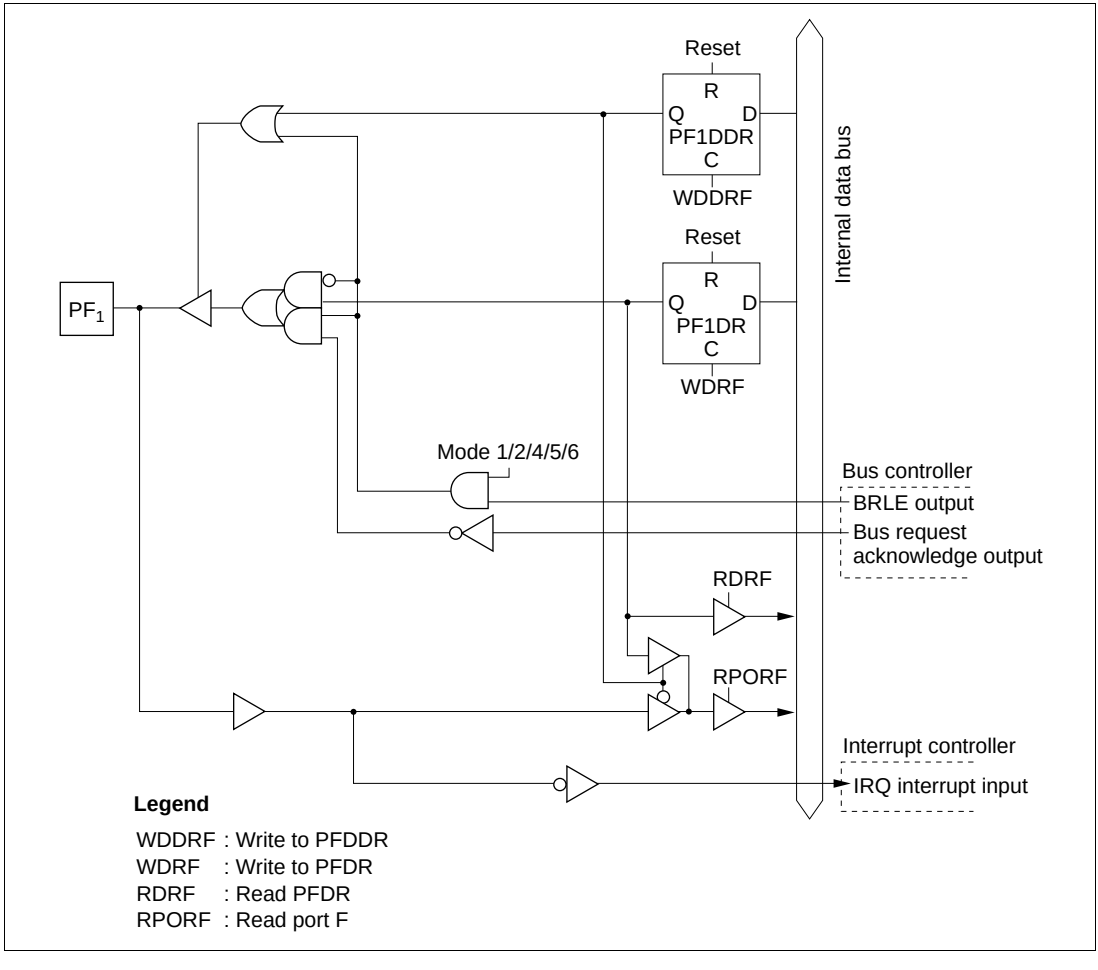



**Figure C.10 (b) Port F Block Diagram (Pin $PF_1$)**

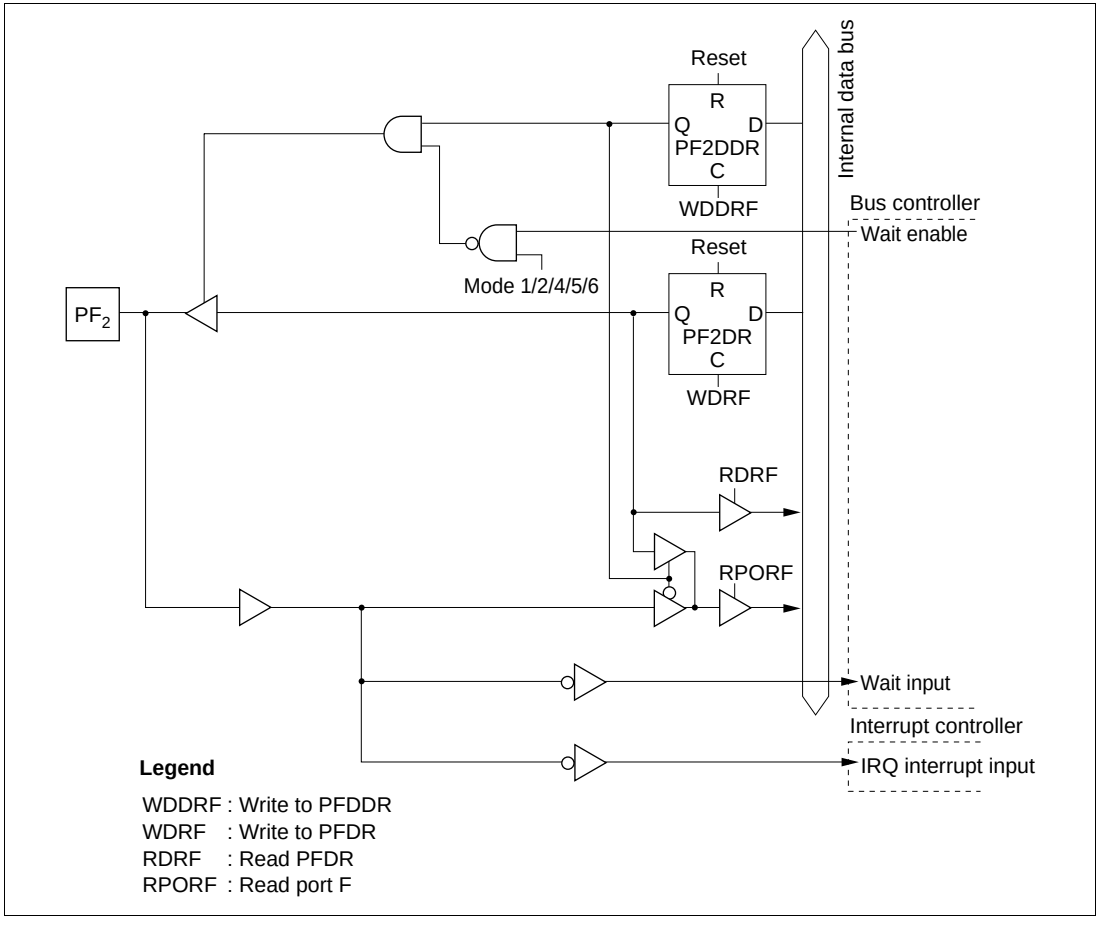



**Figure C.10 (c) Port F Block Diagram (Pin $PF_2$)**

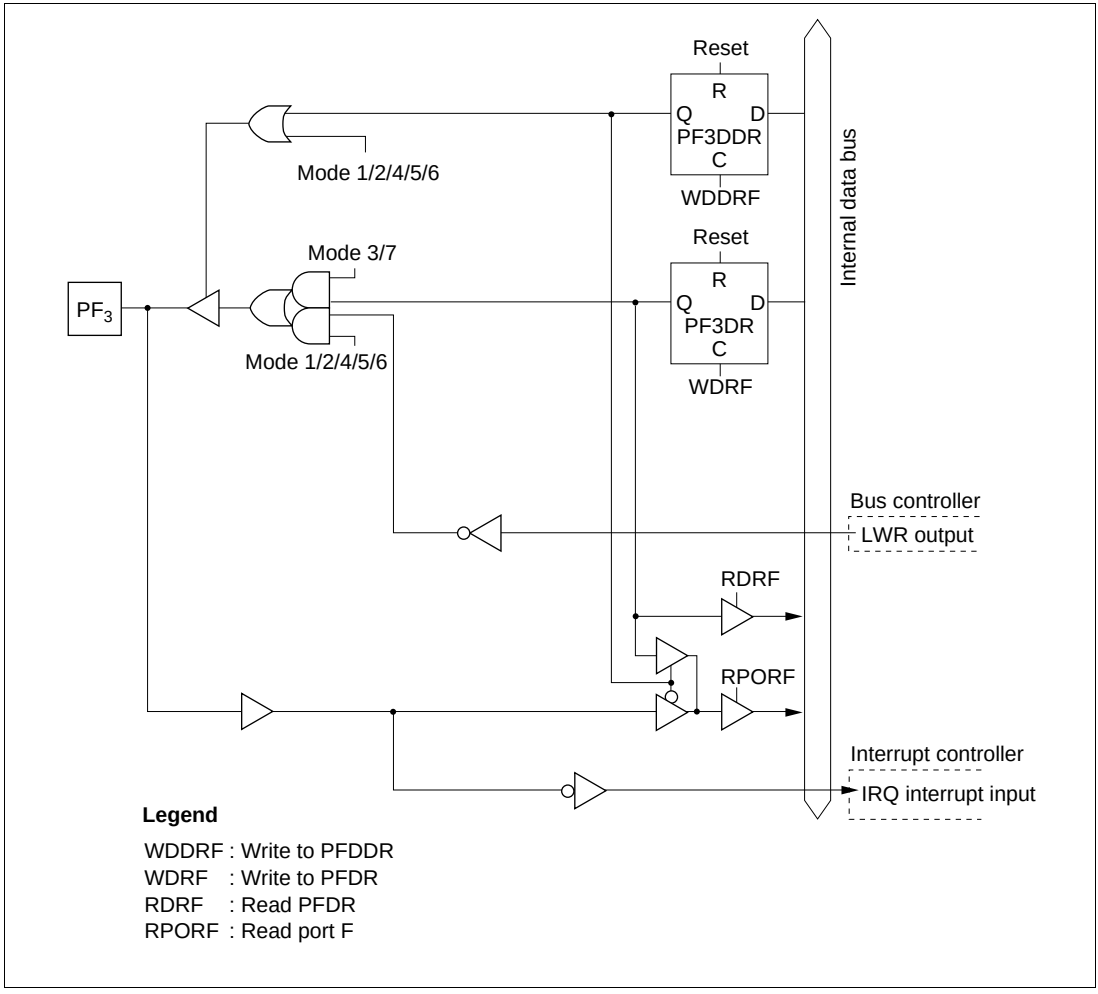



**Figure C.10 (d) Port F Block Diagram (Pin $PF_3$)**

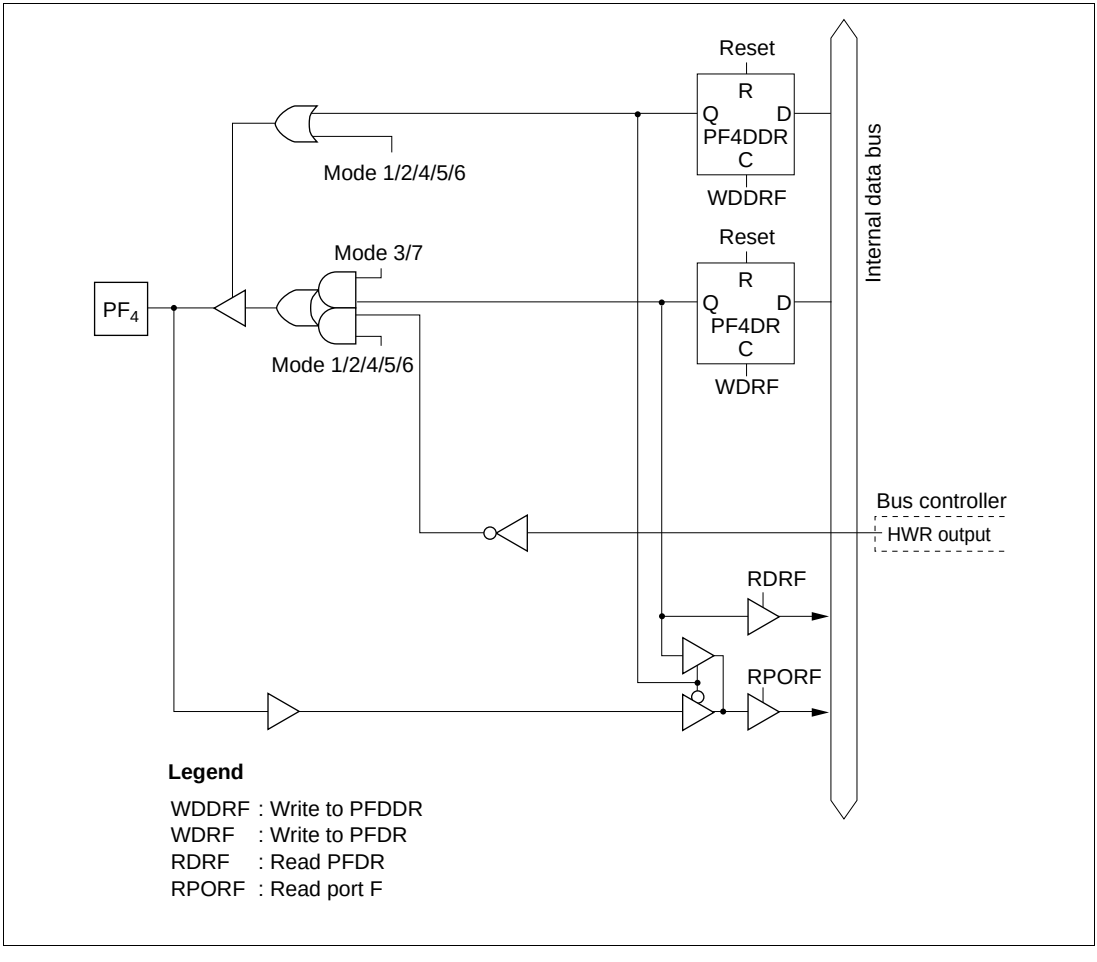



**Figure C.10 (e) Port F Block Diagram (Pin $PF_4$)**

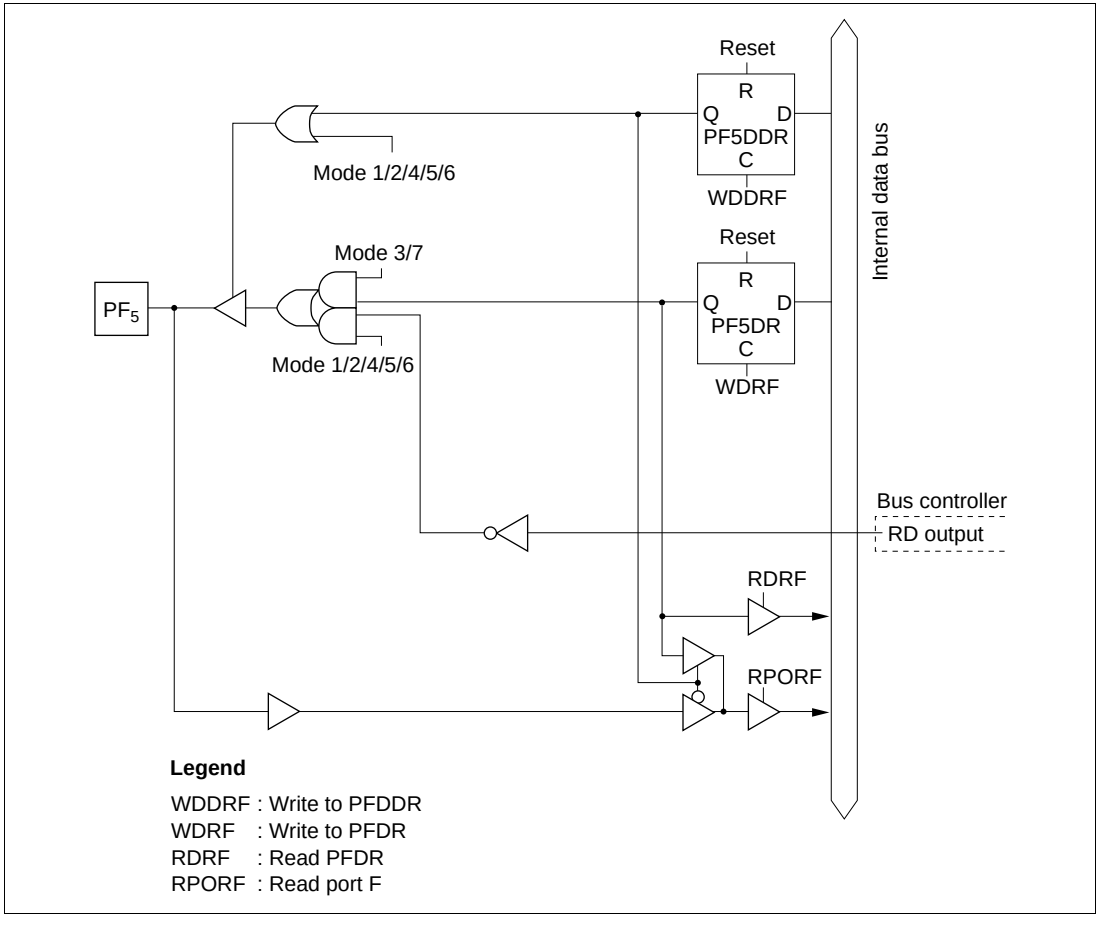



**Figure C.10 (f) Port F Block Diagram (Pin $PF_5$)**

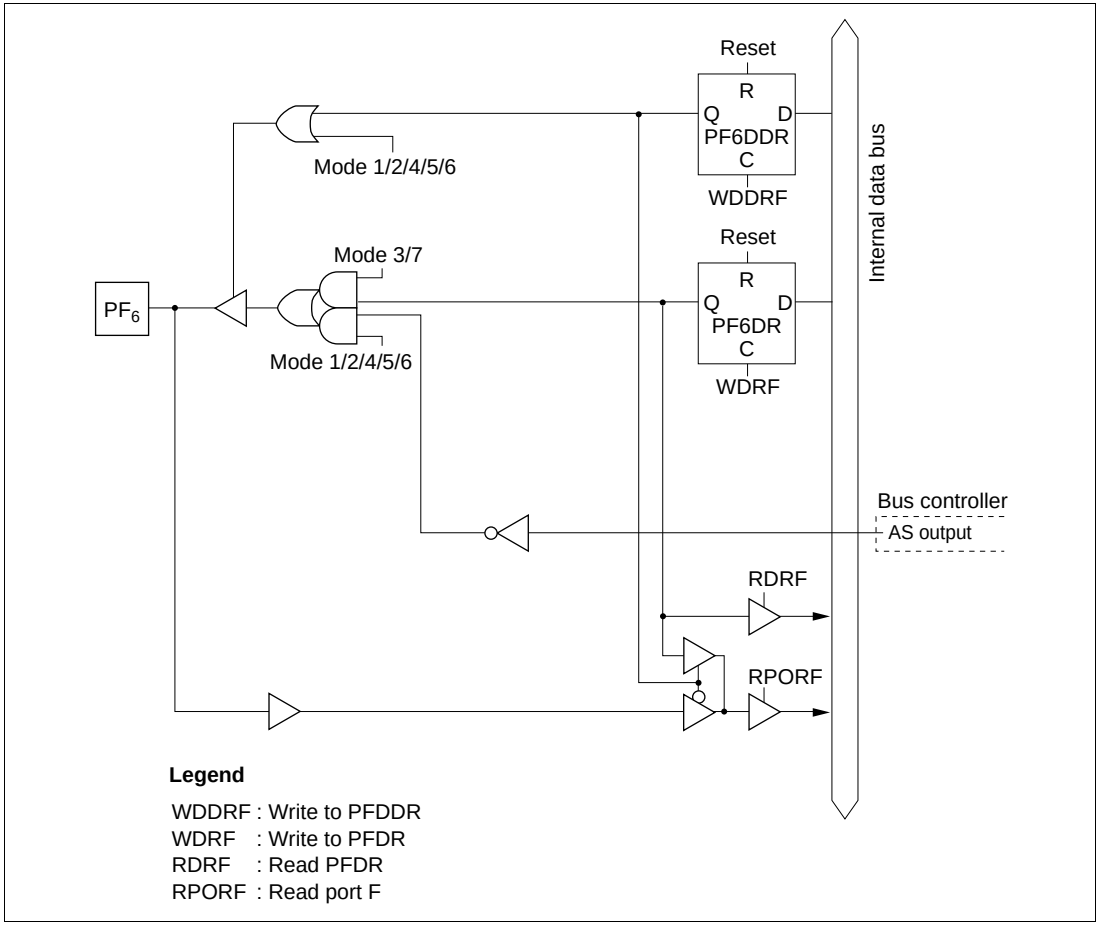



**Figure C.10 (g) Port F Block Diagram (Pin $PF_6$)**

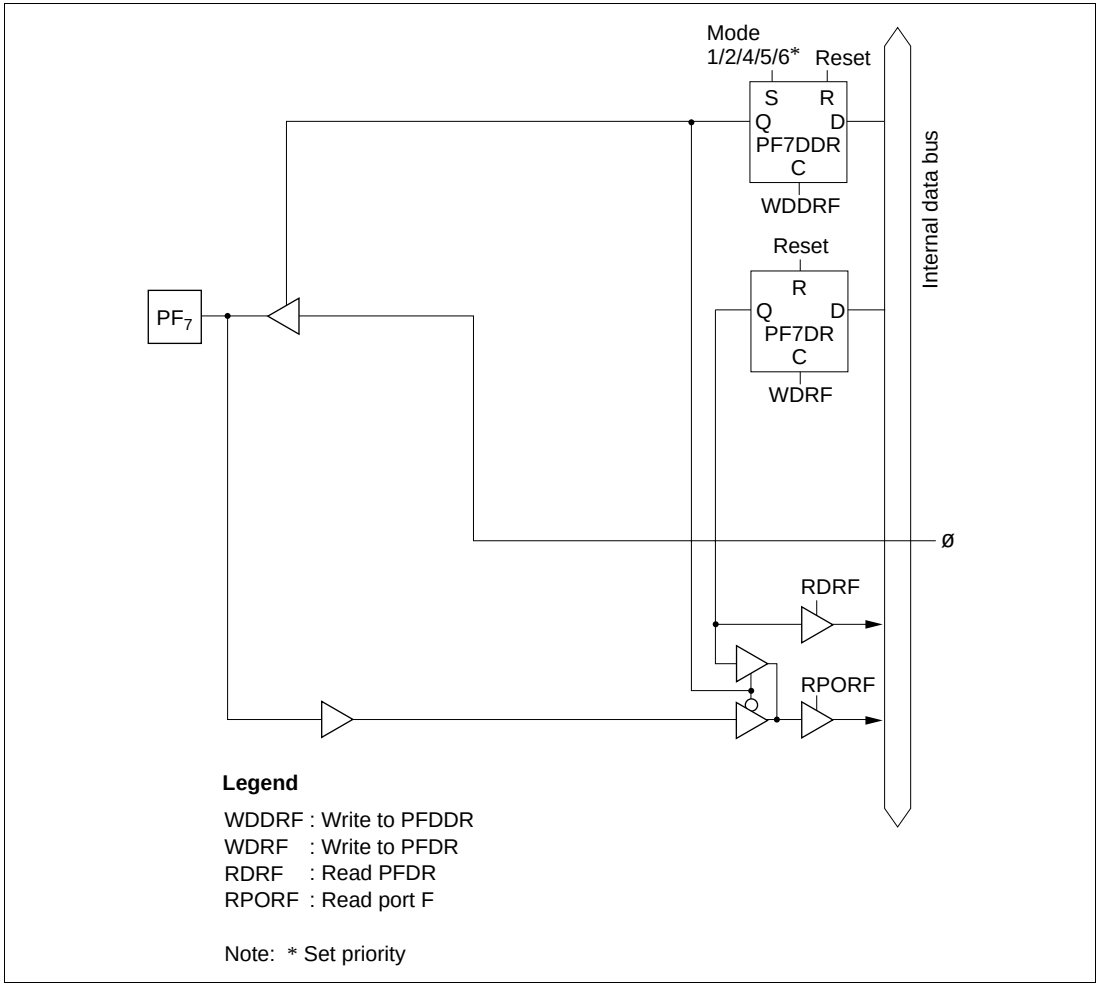



**Figure C.10 (h) Port F Block Diagram (Pin $PF_7$)**

## C.11 Port G Block Diagram

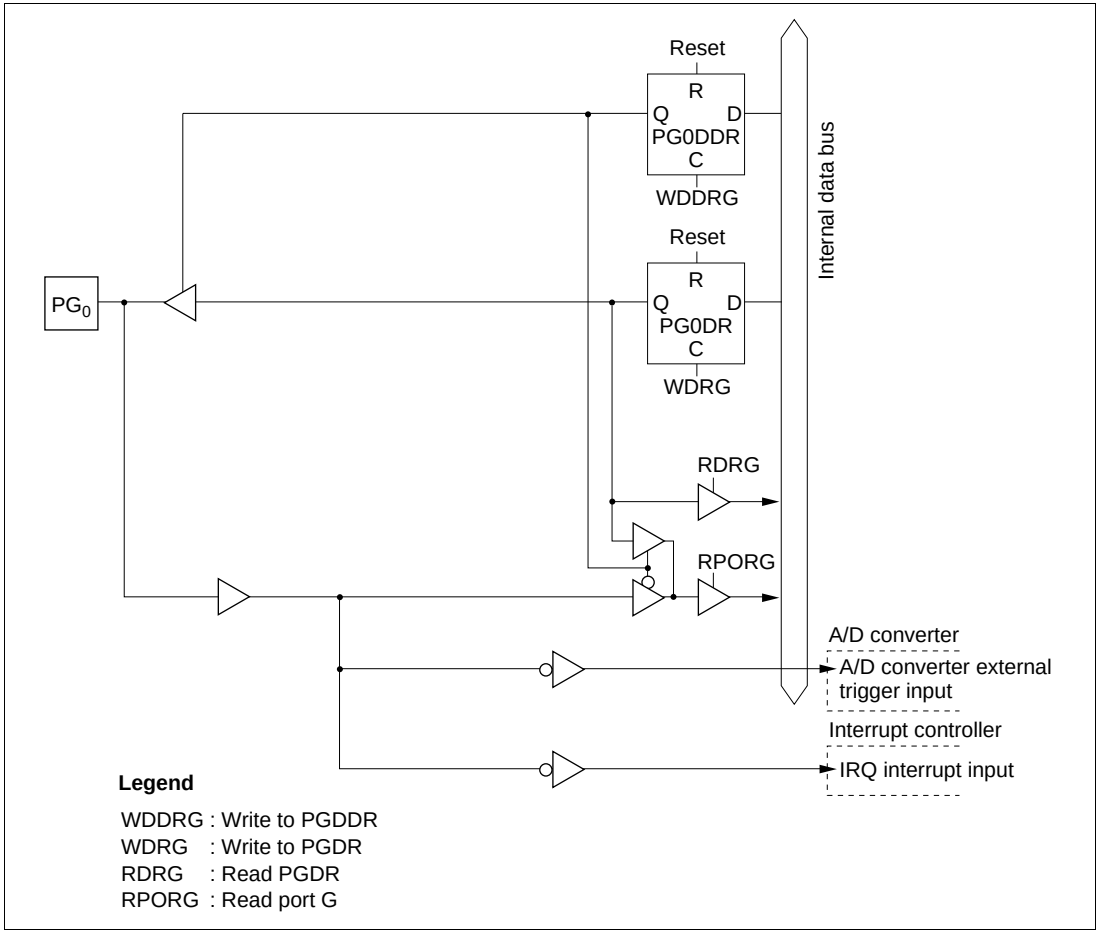



**Figure C.11 (a) Port G Block Diagram (Pin $PG_0$)**

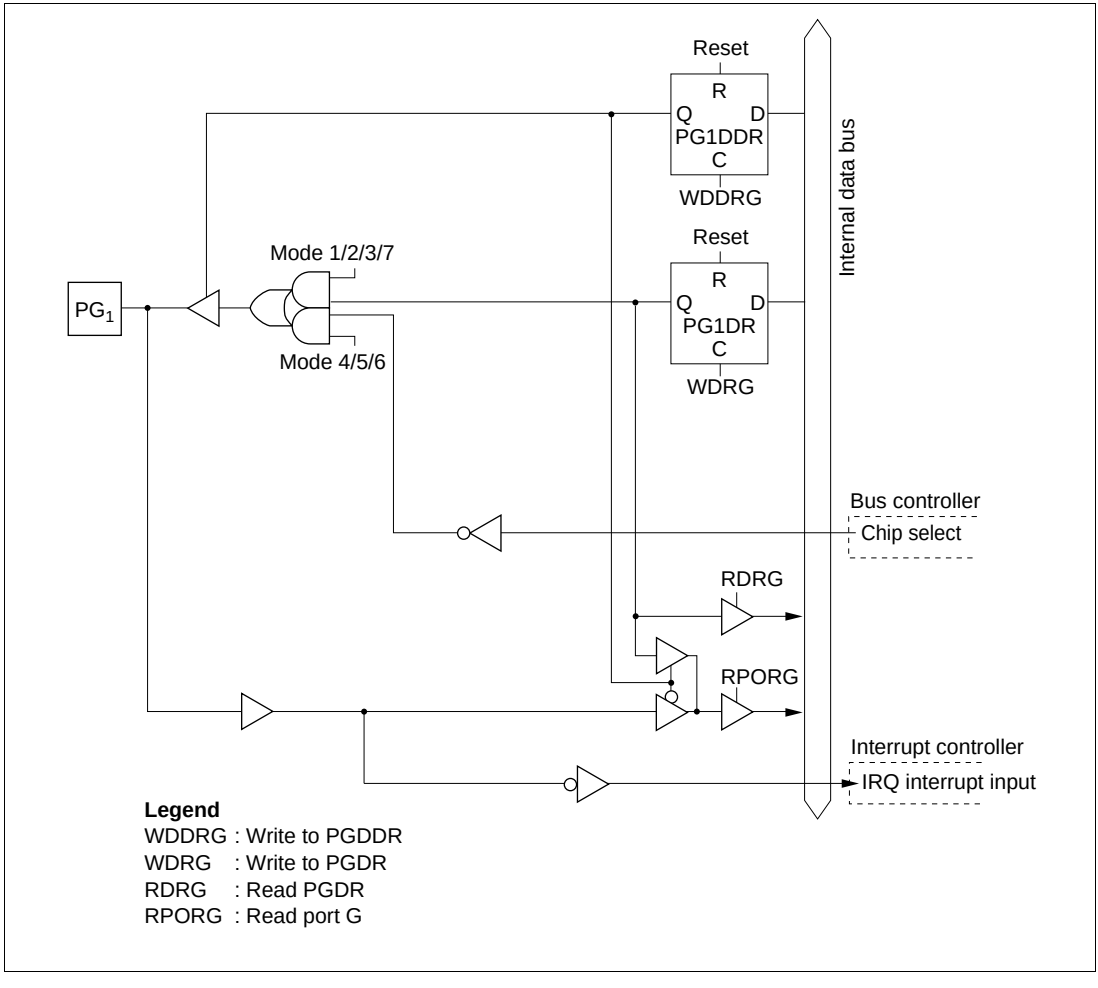



**Figure C.11 (b) Port G Block Diagram (Pin $PG_1$)**

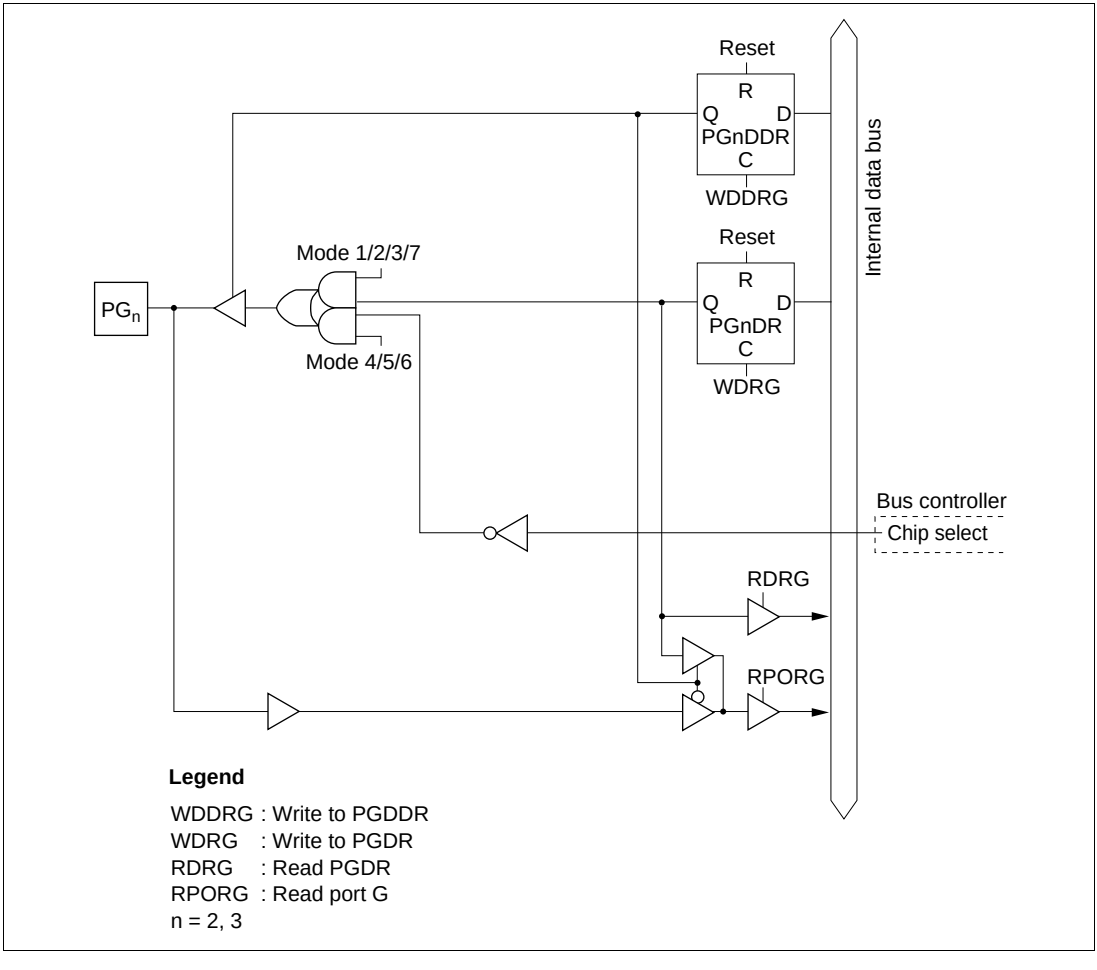



**Figure C.11 (c) Port G Block Diagram (Pins $PG_2$ and $PG_3$)**

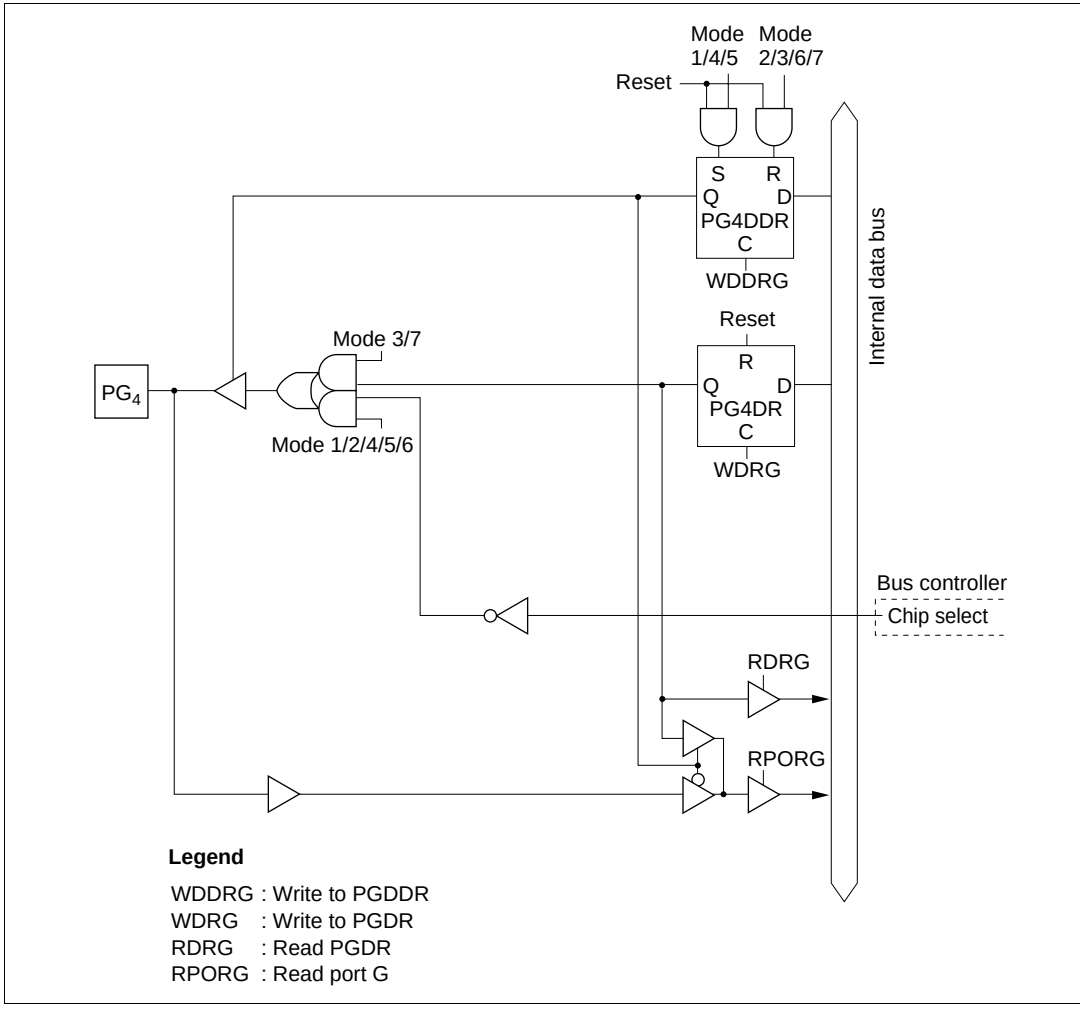



**Figure C.11 (d) Port G Block Diagram (Pin $PG_4$)**

# Appendix D Pin States

## D.1 Port States in Each Mode

### Table D.1 I/O Port States in Each Processing State

| Port Name Pin Name | MCU Operating Mode | Power-On Reset | Manual Reset | Hardware Standby Mode | Software Standby Mode | Bus Release State | Program Execution State Sleep Mode |
|---|---|---|---|---|---|---|---|
| $P1_7$/TIOCB2/ TCLKD $P1_6$/TIOCA2 $P1_5$/TIOCB1/ TCLKC $P1_4$/TIOCA1 | 1 to 7 | T | kept | T | kept | kept | I/O port |
| $P1_3$/TIOCD0/ TCLKB/$A_{23}$ $P1_2$/TIOCC0/ TCLKA/$A_{22}$ $P1_1$/TIOCB0/ $A_{21}$ $P1_0$/TIOCA0/ $A_{20}$ | 1 to 3, 7 | T | kept | T | kept | kept | I/O port |
| | 4 to 6 | T | kept | T | [DDR · OPE = 0] T<br>[DDR · OPE = 1] kept | T | [DDR = 0] Input port<br>[DDR = 1] Address output |
| Port 2 | 1 to 7 | T | kept | T | kept | kept | I/O port |
| Port 3 | 1 to 7 | T | kept | T | kept | kept | I/O port |
| $P4_7$/DA1 | 1 to 7 | T | T | T | [DAOE1 = 1] kept<br>[DAOE1 = 0] T | kept | I/O port |
| $P4_6$/DA0 | 1 to 7 | T | T | T | [DAOE0 = 1] kept<br>[DAOE0 = 0] T | kept | I/O port |
| $P4_5$ to $P4_0$ | 1 to 7 | T | T | T | T | T | Input port |

**Table D.1 I/O Port States in Each Processing State (cont)**

| Port Name Pin Name | MCU Operating Mode | Power-On Reset | Manual Reset | Hardware Standby Mode | Software Standby Mode | Bus Release State | Program Execution State Sleep Mode |
|---|---|---|---|---|---|---|---|
| Port A | 1 to 3, 7 | T | kept | T | kept | kept | I/O port |
| | 4, 5 | L | kept | T | [OPE = 0] T [OPE = 1] kept | T | Address output |
| | 6 | T | kept | T | [DDR · OPE = 0] T [DDR · OPE = 1] kept | T | [DDR = 0] Input port [DDR = 1] Address output |
| Port B | 1, 4, 5 | L | kept | T | [OPE = 0] T [OPE = 1] kept | T | Address output |
| | 2, 6 | T | kept | T | [DDR · OPE = 0] T [DDR · OPE = 1] kept | T | [DDR = 0] Input port [DDR = 1] Address output |
| | 3, 7 | T | kept | T | kept | kept | I/O port |
| Port C | 1, 4, 5 | L | kept | T | [OPE = 0] T [OPE = 1] kept | T | Address output |
| | 2, 6 | T | kept | T | [DDR · OPE = 0] T [DDR · OPE = 1] kept | T | [DDR = 0] Input port [DDR = 1] Address output |
| | 3, 7 | T | kept | T | kept | kept | I/O port |
| Port D | 1, 2, 4 to 6 | T | T* | T | T | T | Data bus |
| | 3, 7 | T | kept | T | kept | kept | I/O port |
| Port E | 1, 2, 4 to 6 (8 bit bus) | T | kept | T | kept | kept | I/O port |
| | (16 bit bus) | T | T* | T | T | T | Data bus |
| | 3, 7 | T | kept | T | kept | kept | I/O port |

**Table D.1 I/O Port States in Each Processing State (cont)**

| Port Name Pin Name | MCU Operating Mode | Power-On Reset | Manual Reset | Hardware Standby Mode | Software Standby Mode | Bus Release State | Program Execution State Sleep Mode |
|---|---|---|---|---|---|---|---|
| $PF_7$/ø | 1, 2, 4 to 6 | Clock output | [DDR = 0] T [DDR = 1] Clock output | T | [DDR = 0] Input port [DDR = 1] H | [DDR = 0] Input port [DDR = 1] Clock output | [DDR = 0] Input port [DDR = 1] Clock output |
| | 3, 7 | T | kept | T | [DDR = 0] Input port [DDR = 1] H | [DDR = 0] Input port [DDR = 1] Clock output | [DDR = 0] Input port [DDR = 1] Clock output |
| $PF_6$/$\overline{AS}$ $PF_5$/$\overline{RD}$ $PF_4$/$\overline{HWR}$ $PF_3$/$\overline{LWR}$/ $\overline{IRQ3}$ | 1, 2, 4 to 6 | H | H* | T | [OPE = 0] T [OPE = 1] H | T | $\overline{AS}$, $\overline{RD}$, $\overline{HWR}$, $\overline{LWR}$ |
| | 3, 7 | T | kept | T | kept | kept | I/O port |
| $PF_2$/$\overline{WAIT}$/ $\overline{IRQ2}$ | 1, 2, 4 to 6 | T | [WAITE = 0] kept [WAITE = 1] T | T | [WAITE = 0] kept [WAITE = 1] T | [WAITE = 0] kept [WAITE = 1] T | [WAITE = 0] I/O port [WAITE = 1] $\overline{WAIT}$ |
| | 3, 7 | T | kept | T | kept | kept | I/O port |
| $PF_1$/$\overline{BACK}$/ $\overline{IRQ1}$ | 1, 2, 4 to 6 | T | [BRLE = 0] kept [BRLE = 1] $\overline{BACK}$ | T | [BRLE = 0] kept [BRLE = 1] H | L | [BRLE = 0] I/O port [BRLE = 1] $\overline{BACK}$ |
| | 3, 7 | T | kept | T | kept | kept | I/O port |
| $PF_0$/$\overline{BREQ}$/ $\overline{IRQ0}$ | 1, 2, 4 to 6 | T | [BRLE = 0] kept [BRLE = 1] $\overline{BREQ}$ | T | [BRLE = 0] kept [BRLE = 1] T | T | [BRLE = 0] I/O port [BRLE = 1] $\overline{BREQ}$ |
| | 3, 7 | T | kept | T | kept | kept | I/O port |
| $PG_4$/$\overline{CS0}$ | 1, 4, 5 | H | [DDR = 0] T [DDR = 1] H* | T | [DDR · OPE = 0] T [DDR · OPE = 1] H | T | [DDR = 0] Input port [DDR = 1] $\overline{CS_0}$ |
| | 2, 6 | T | | | | | |
| | 3, 7 | T | kept | T | kept | kept | I/O port |

**Table D.1 I/O Port States in Each Processing State (cont)**

| Port Name Pin Name | MCU Operating Mode | Power-On Reset | Manual Reset | Hardware Standby Mode | Software Standby Mode | Bus Release State | Program Execution State Sleep Mode |
|---|---|---|---|---|---|---|---|
| $PG_3/\overline{CS1}$ | 1 to 3, 7 | T | kept | T | kept | kept | I/O port |
| $PG_2/\overline{CS2}$ $PG_1/\overline{CS3}/\overline{IRQ7}$ | 4 to 6 | T | [DDR = 0] T [DDR = 1] H* | T | [DDR · OPE = 0] T [DDR · OPE = 1] H | T | [DDR = 0] Input port [DDR = 1] $\overline{CS}_1$ to $\overline{CS}_3$ |
| $\overline{PG_0}/\overline{ADTRG}/\overline{IRQ6}$ | 1 to 3, 7 | T | kept | T | kept | kept | I/O port |
| | 4 to 6 | T | kept | T | kept | T | I/O port |

**Legend:**

H : High level
L : Low level
T : High impedance
kept : Input port becomes high-impedance, output port retains state
DDR : Data direction register
OPE : Output port enable
WAITE : Wait input enable
BRLE : Bus release enable

Note: * Indicates the state after completion of the executing bus cycle.

# Appendix E Timing of Transition to and Recovery from Hardware Standby Mode

## Timing of Transition to Hardware Standby Mode

(1) To retain RAM contents with the RAME bit set to 1 in SYSCR, drive the $\overline{RES}$ signal low at least 10 states before the $\overline{STBY}$ signal goes low, as shown below. $\overline{RES}$ must remain low until $\overline{STBY}$ signal goes low (delay from $\overline{STBY}$ low to $\overline{RES}$ high: 0 ns or more).

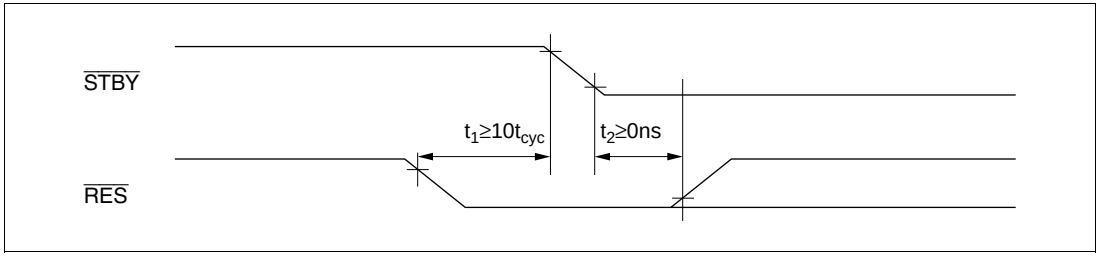



**Figure E.1 Timing of Transition to Hardware Standby Mode**

(2) To retain RAM contents with the RAME bit cleared to 0 in SYSCR, or when RAM contents do not need to be retained, $\overline{RES}$ does not have to be driven low as in (1).

## Timing of Recovery from Hardware Standby Mode

Drive the $\overline{RES}$ signal low and the NMI signal high approximately 100 ns or more before $\overline{STBY}$ goes high to execute a power-on reset.

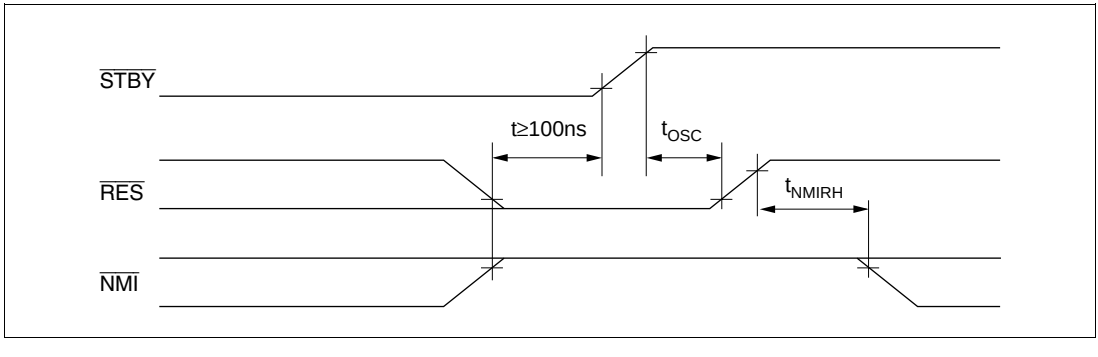



**Figure E.2 Timing of Recovery from Hardware Standby Mode**

# Appendix F Product Code Lineup

**Table F.1 H8S/2345 Series Product Code Lineup**

| Product Type | | Product Code | Mark Code | Package (Hitachi Package Code) |
|---|---|---|---|---|
| H8S/2345 | Mask ROM | HD6432345 | HD6432345(***)TE | 100-pin TQFP (TFP-100B) |
| | | | HD6432345(***)TF | 100-pin TQFP (TFP-100G)* |
| | | | HD6432345(***)F | 100-pin QFP (FP-100A) |
| | | | HD6432345(***)FA | 100-pin QFP (FP-100B) |
| | ZTAT™ | HD6472345 | HD6472345TE | 100-pin TQFP (TFP-100B) |
| | | | HD6472345TF | 100-pin TQFP (TFP-100G)* |
| | | | HD6472345F | 100-pin QFP (FP-100A) |
| | | | HD6472345FA | 100-pin QFP (FP-100B) |
| | F-ZTAT™ | HD64F2345 | HD64F2345TE | 100-pin TQFP (TFP-100B) |
| | | | HD64F2345TF | 100-pin TQFP (TFP-100G)* |
| | | | HD64F2345F | 100-pin QFP (FP-100A) |
| | | | HD64F2345FA | 100-pin QFP (FP-100B) |
| H8S/2344 | Mask ROM | HD6432344 | HD6432344(***)TE | 100-pin TQFP (TFP-100B) |
| | | | HD6432344(***)TF | 100-pin TQFP (TFP-100G)* |
| | | | HD6432344(***)F | 100-pin QFP (FP-100A) |
| | | | HD6432344(***)FA | 100-pin QFP (FP-100B) |
| H8S/2343 | Mask ROM | HD6432343 | HD6432343(***)TE | 100-pin TQFP (TFP-100B) |
| | | | HD6432343(***)TF | 100-pin TQFP (TFP-100G)* |
| | | | HD6432343(***)F | 100-pin QFP (FP-100A) |
| | | | HD6432343(***)FA | 100-pin QFP (FP-100B) |
| H8S/2341 | Mask ROM | HD6432341 | HD6432341(***)TE | 100-pin TQFP (TFP-100B) |
| | | | HD6432341(***TF | 100-pin TQFP (TFP-100G)* |
| | | | HD6432341(***)F | 100-pin QFP (FP-100A) |
| | | | HD6432341(***)FA | 100-pin QFP (FP-100B) |
| H8S/2340 | ROMless | HD6412340 | HD6412340TE | 100-pin TQFP (TFP-100B) |
| | | | HD6412340TF | 100-pin TQFP (TFP-100G)* |
| | | | HD6412340F | 100-pin QFP (FP-100A) |
| | | | HD6412340FA | 100-pin QFP (FP-100B) |

Note: (***) indicates the ROM code.

* TFP-100G is under development.

# Appendix G Package Dimensions

Figures G.1 to G.4 show the package dimensions of the H8S/2345 Series.

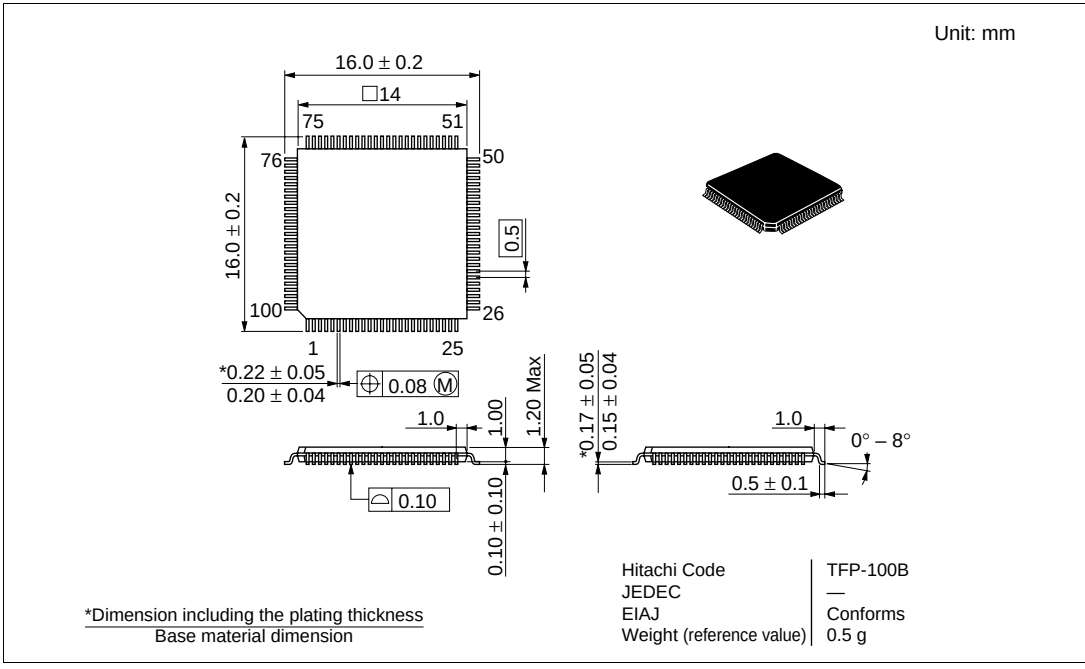
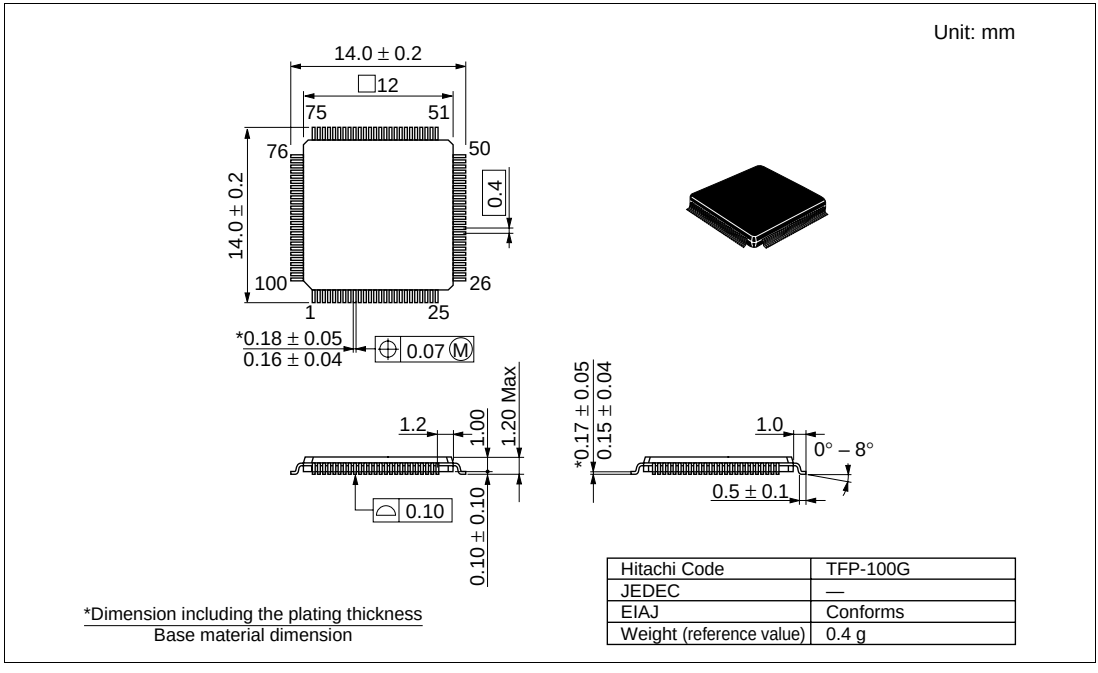



**Figure G.1 TFP-100B Package Dimensions**




| | |
|---|---|
| Hitachi Code | TFP-100G |
| JEDEC | — |
| EIAJ | Conforms |
| Weight (reference value) | 0.4 g |

**Figure G.2 TFP-100G Package Dimensions (Under development)**

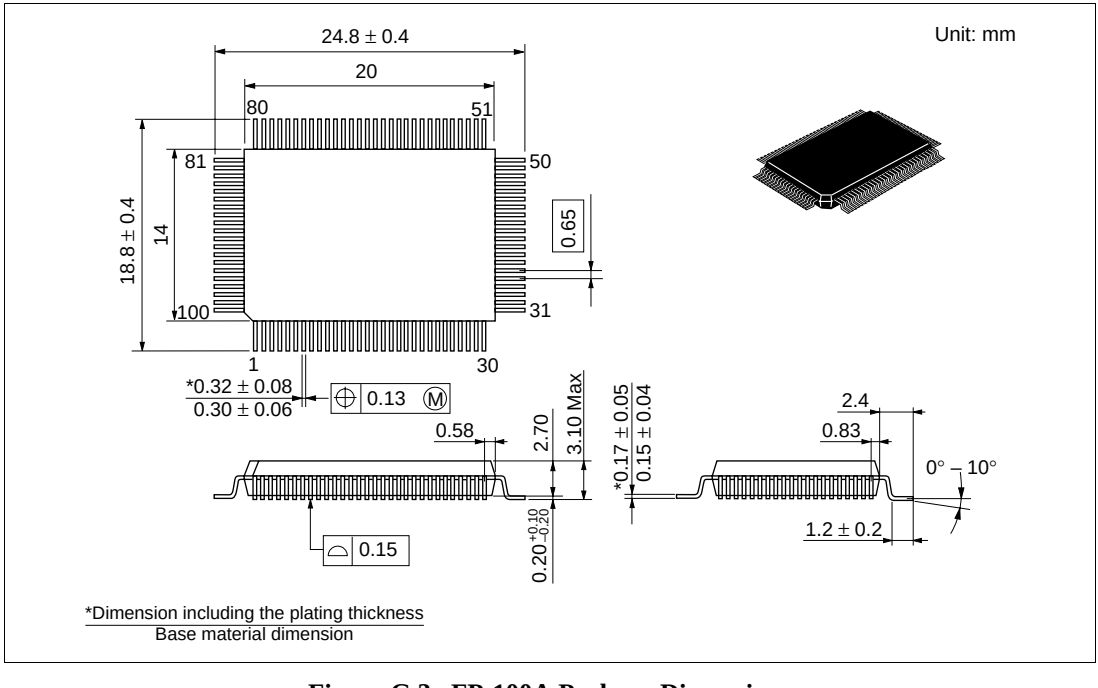



**Figure G.3 FP-100A Package Dimensions**

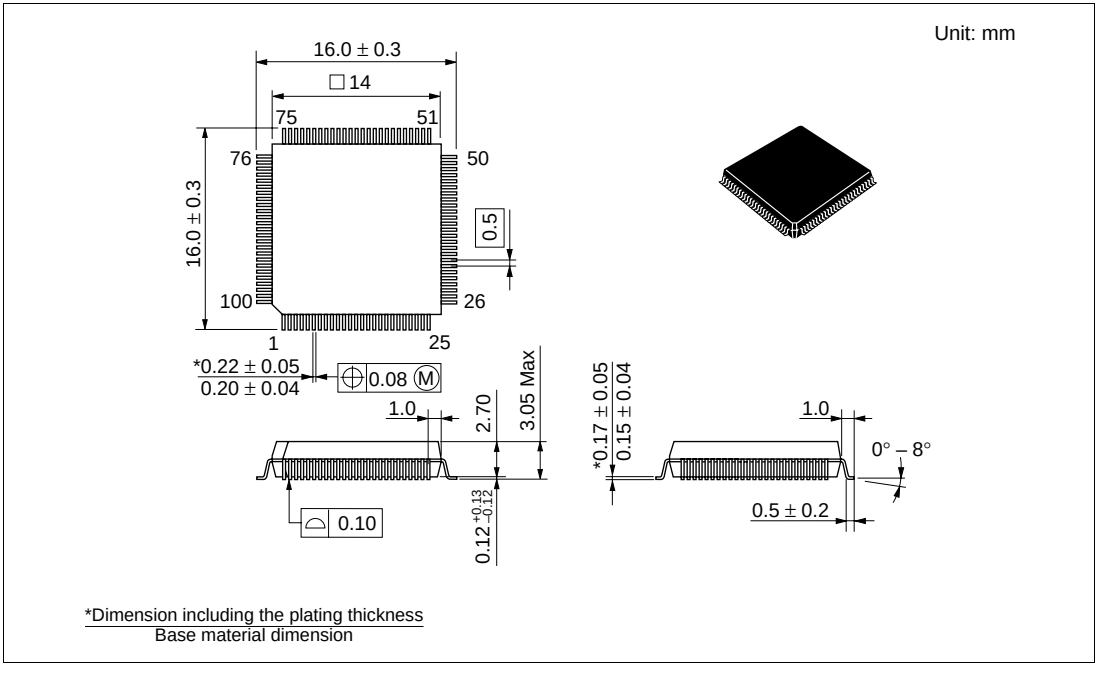



**Figure G.4 FP-100B Package Dimensions**

**H8S/2345 Series, H8S/2345F-ZTAT™**
**Hardware Manual**

Publication Date: 1st Edition, September 1997
2nd Edition, December 1998
Published by: Electronic Devices Sales & Marketing Group
Semiconductor & Integrated Circuits Group
Hitachi, Ltd.
Edited by: Technical Documentation Group
UL Media Co., Ltd.
Copyright © Hitachi, Ltd., 1997. All rights reserved. Printed in Japan.